APLICACIÓ BIOINFORMÀTICA PER AL DESCOBRIMENT DE NOUS FÀRMACS

Marcel Otón Pàmies

8 de Novembre de 2013

Títol: “Aplicació bioinformàtica per al descobriment de nous fàrmacs”

Titulació: Enginyeria Informàtica

Centre: Facultat d’Informàtica de Barcelona

Universitat: Universitat Politècnica de Catalunya

Data: 8 de Novembre de 2013

Autor: Marcel Otón Pàmies

Director: Ramón Goñi Macià

Ponent: Toni Cortés Roselló

Presidenta: Rosa Maria Badia Sala

Vocal: Rosa Maria Jiménez Gómez

Índex

0. Convencions i notació utilitzada

En la redacció d'aquest document es faran servir un seguit de convencions, detallades a continuació, per tal de facilitar la comprensió del text al lector.

En primer lloc, algunes de les paraules tècniques, així com els noms d'algunes de les tecnologies emprades, es mantindran en anglès. Això es fa per tal d'assimilar el llenguatge de la memòria amb el llenguatge que es fa servir habitualment. Així doncs, paraules com *scripts* (traduït textualment com a guió) es mantindran en anglès ja que la paraula original resultarà més familiar al lector o si més no l'ajudaran a trobar descripcions. Aquestes paraules es representaran en cursiva per tal d'informar al lector d'aquesta particularitat.

Per altra banda, quan s'hagi d'explicar algun concepte per tal de facilitar la comprensió del text, es faran servir dos mètodes: els annexos i la bibliografia. Quan algun concepte estigui explicat en algun annex s'especificarà explícitament l'annex on queda detallat. Quan un concepte queda explicat en alguna entrada de la bibliografia utilitzada es fa servir el subíndex concepte $_x$, on x és el número de l'entrada de la bibliografia on es fa referència al concepte.

1. Context

El següent projecte ha estat desenvolupat dintre d'un grup del departament de Life Science del Barcelona Supercomputing Center - Centre Nacional de Supercomputació (BSC - CNS). El BSC és un centre de supercomputació ubicat a Barcelona que allotja recerca en l'àrea de la computació, les ciències de la terra i les ciències de la vida. Un dels principals recursos del centre és el supercomputador MareNostrum III, amb una capacitat de càlcul teòrica d'1Pflop. A més d'oferir capacitat de càlcul per als investigadors de la pròpia institució, el BSC també proporciona aquesta capacitat a altres centres de recerca o institucions privades.

En el departament de Life Sciences es duen a terme diferents projectes d'investigació, tots orientats al camp de la salut. En aquests projectes s'intenta trobar respostes a problemes fisiològics mitjançant simulacions per ordinador. Els principals beneficis dels mètodes computacionals (respecte als purament experimentals) són: una reducció en el temps emprat per processar les dades i un menor cost econòmic.

En el següent document s'explicarà detalladament el procés que s'ha seguit per desenvolupar una eina bioinformàtica en aquest departament. Primerament, es donarà una breu descripció dels objectius del projecte. Posteriorment s'estudiarà amb més detall l'aplicació: s'analitzaran els requisits (funcionals i no funcionals) i casos d'ús, i es donarà un disseny conceptual d'aquesta. Tot seguit es procedirà a explicar els passos seguits en el disseny de l'eina: es mostrarà l'estructura interna de l'aplicació, com es relacionen els diferents components entre si i el procediment que s'ha seguit a l'hora de desenvolupar-la. En un altre apartat es veuran els passos que s'han seguit en la implementació de l'aplicació, així com les diferències que han anat sorgint des del primer esbós fins a la finalització del projecte. Per altra banda, hi haurà un apartat on es parlarà de possibles millores en el futur per a l'aplicació, que no s'han pogut realitzar en aquest projecte però que incrementaran la qualitat general de l'eina si són implementades. A part, trobarem un manual d'usuari de l'aplicació, on s'explicarà detalladament el funcionament de l'eina i es podran veure totes les funcionalitats desenvolupades. Finalment hi haurà la bibliografia i els annexos, utilitzats per tal de donar més informació sobre conceptes que no formen part

explícita de l’aplicació però sense el coneixement dels quals es perdria part de l’abast del projecte.

2.Introducció

2.1. Motivació

Un dels grans beneficis de la informàtica és que pot ser aplicada en una gran varietat de camps per tal de solucionar problemes molt diferents. Aquest ha estat un dels factors que van influir a l'hora d'escollir aquest com al meu Projecte de Final de Carrera. D'aquesta manera, he sigut capaç de desenvolupar una eina fent servir les metodologies apreses durant tota la carrera, en un camp totalment nou per a mi com la biologia.

Tot i que pugui semblar que biologia i informàtica estiguin poc relacionades donat que intenten resoldre problemes diferents, la realitat és ben diferent: actualment, i cada vegada més, els biòlegs utilitzen eines i solucions informàtiques per tal de trobar una resposta als seus problemes quotidians. D'aquesta relació en sorgeix la bioinformàtica[1], especialitat interdisciplinària que utilitza i desenvolupa eines informàtiques per tal de generar, emmagatzemar, organitzar i analitzar dades biològiques.

La motivació per realitzar un PFC en aquest camp ve motivada principalment per l'ús aplicat de la informàtica i per com d'ell en sorgeixen resultats que ajuden a millorar la vida de les persones mitjançant tècniques que no eren possibles fins ara.

2.2. El projecte

El programari bioinformàtic és una eina indispensable en el procés de descobriment de nous fàrmacs. La bioinformàtica té molts camps d'aplicació i un d'ells és la predicció i simulació *in silico* (per computador) de l'activitat entre una petita molècula (fàrmac candidat) i la seva diana terapèutica (normalment una proteïna).

El meu projecte de final de carrera consisteix en la implementació d'una eina informàtica anomenada SEABED (Small molEcule Activity scanner web-service Based on Ensemble Docking). Aquesta eina web serà utilitzada per la predicció ràpida i de baix cost econòmic d'interaccions entre proteïnes i petites molècules. L'interès de fer prediccions de baix cost és escanejar i reduir llibreries amb milions de compostos a subgrups de milers enriquits amb molècules amb dues propietats: propensió a tenir activitat contra la diana

seleccionada i que siguin molècules més representatives de tot el conjunt. Reduint el volum de les llibreries de molècules (de milions a milers) farem factible el cribratge experimental per identificar aquelles que mostren activitat farmacològica.

L'objectiu principal del projecte és dissenyar i implementar una eina web orientada a usuaris no experts en bioinformàtica. Els algoritmes bioinformàtics per la simulació i predicció de l'activitat entre proteïnes i petites molècules han estat desenvolupats prèviament en el grup de recerca del BSC o per tercers, i estan implementats en llibreries *scripts* de difícil accés sense una expertesa prèvia en el camp.

Amb la nova eina web, els usuaris no experts en bioinformàtica (biòlegs, químics, etc.) tindran un accés fàcil a la tecnologia d'enriquiment de compostos químics per als seus projectes. Per adaptar-se millor a cada necessitat, la web permet dos tipus d'accés: l'accés anònim o l'accés amb compte d'usuari. Per projectes petits o de curt termini, l'usuari podrà treballar de manera privada però anònima, amb una sessió temporal que no requereixi cap tipus d'identificació. Per projectes de mig-llarg termini, l'eina ofereix la possibilitat de registrar-se i crear un compte personal on podrà crear projectes independents.

Per altra banda, degut a que les simulacions requereixen d'un elevat nombre de càlculs i treballen amb fitxers grans (fins a *gigabytes*), l'execució dels diferents processos es pot demorar algunes hores depenent del cas. Per tal de gestionar els processos que executen els usuaris, i per tal que aquests puguin treballar en un o més *workflows* al mateix temps, s'ha utilitzat un sistema de cues per a l'execució dels càlculs que requereixen d'un temps més elevat per executar-se. Es complimenta aquest sistema de cues amb un servei de notificacions, que alertarà als usuaris que ho desitgin quan han acabat d'executar-se els *scripts* que requereixen més temps.

En una segona iteració del projecte, s'ha dissenyat un model de dades per a l'aplicació SEABED. Posteriorment, s'ha traduït aquest model a una base de dades relacional que s'ha integrat a l'aplicació. Gràcies a aquesta integració, s'ha aconseguit gestionar les dades d'una manera més eficient i a més a més es permet poder realitzar mineria de dades sobre la informació gestionada en un futur.

2.2.1. Workflows

Mitjançant l'aplicació, els usuaris podran realitzar diversos *workflows*. Entenem com a *workflow* el procés que segueix un usuari per tal d'obtenir un resultat a partir d'unes dades inicials i després d'aplicar un seguit d'operacions sobre aquestes. Es podran realitzar els següents:

- *Dockings* [1]
- Creació de models predictius utilitzant la metodologia *Quantitative Structure–Activity Relationship* [2] (QSAR)
- Creació de models predictius utilitzant la metodologia *Pseudo-Quantitative Structure–Activity Relationship* [3,4] (PSAR)
- Aplicació de models predictius a dades de test

2.2.1.1. Dockings

En el camp de la biologia molecular, el docking fa referència a la predicció de la orientació òptima entre dues molècules quan s'uneixen per a formar un compost. Mitjançant aquesta tècnica i funcions de puntuació es pot quantificar l'afinitat que tenen dues molècules quan s'uneixen. La zona de la proteïna on es formen els enllaços amb la molècula s'anomena *binding site.*

Aquest mètode s'utilitza en el camp del descobriment de nous fàrmacs per tal de trobar molècules (o fàrmacs potencials) que generin activitat quan s'uneixen a proteïnes (o dianes terapèutiques). Aquesta activitat es la que permet que, quan es forma el compost, l'efecte no desitjat de la proteïna es vegi reduït o eliminat.

Actualment es fan servir dos tipus de *dockings* per tal de predir aquesta afinitat: rígid i elàstic, cadascun amb els seus avantatges i inconvenients. Quan s'utilitza el *docking* rígid s'intenta unir proteïna i molècula en els punts on la fisiologia d'aquestes es complementària. Aquest mètode es anàleg al joc del *puzzle*, amb l'afegit que un enllaç entre les peces pot ser quantificat depenent de l'espai que queda lliure entre les dues peces. El principal avantatge d'aquest mètode davant del *docking* elàstic és que es pot

calcular molt ràpidament, però per contra no representa tant fidelment la realitat, on les proteïnes i les molècules tenen una estructura variable.

El *docking* elàstic és una via alternativa al *docking* rígid. En aquest cas, les estructures de les proteïnes i les molècules van sofrint lleugers canvis. Això permet reflexar més fidelment la realitat, i més concretament el que es coneix com reconeixement molecular, és a dir, els canvis que sofreixen les dues estructures quan una proteïna i una molècula estan molt properes en l'espai.

Aquests canvis fisiològics estan relacionats amb les seves coordenades, i es poden aconseguir simulant rotacions o translacions. No obstant, aquestes simulacions necessiten uns càlculs molt complexos, que requereixen de grans computadors per ser processades en un temps raonable.

En el cas de l'aplicació SEABED, els usuaris podran realitzar dockings dels dos tipus. En tots dos casos l'usuari haurà de proporcionar les molècules i la proteïna a la qual es vol unir. Com a resultat, es mostraran les afinitats de cadascuna de les molècules amb la proteïna en qüestió.

En el *docking* elàstic, s'introdueix el concepte de *snapshot*. Un *snapshot* d'una proteïna és una instància d'aquesta proteïna en un moment del temps i una trajectòria determinats. Així doncs, una proteïna pot tenir múltiples instàncies en una trajectòria, amb la única diferència que cadascuna d'aquestes instàncies tindran unes coordenades que la faran única en la trajectòria.

Per tal de realitzar el *docking* elàstic, doncs, es farà una simulació prèvia partint d'una proteïna i una trajectòria inicials, s'obtindran un nombre de *snapshots* de la proteïna, i posteriorment es realitzarà el *docking* amb les molècules. Com es pot veure, el procediment serà igual que en el *docking* rígid amb la diferència que s'haurà d'executar tantes vegades com el nombre de *snapshots* que s'agafin. Un cop realitzats tots els *dockings*, l'usuari veurà quina és la millor puntuació i el valor mitjà de les puntuacions dels *snapshots* per a cada molècula. Això es diferencia del cas anterior, en el que només es feia un *docking*, ja que llavors només es donava la millor puntuació (llavors no tenia sentit proporcionar el valor mitjà ja que només existia un valor).

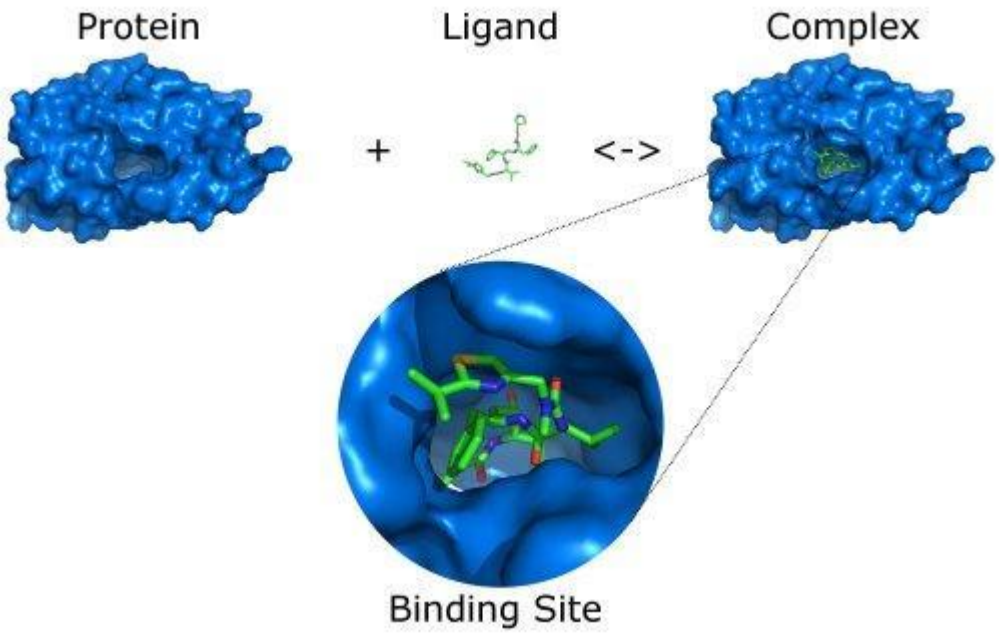


Figura : Diagrama de docking entre una proteïna (proteasa HIV-1) i un lligand (fàrmac Ritonavir)

Per tal de quantificar l'activitat, els resultats de *docking* venen determinats per l'energia que es desprèn de l'enllaç entre les molècules i les proteïnes. Aquesta energia es dóna en kcal/mol. Depenent d'aquest valor es poden fer estimacions de l'afinitat entre els fàrmacs i les dianes terapèutiques: com més negatiu és el valor, més fort serà l'enllaç. Empíricament s'ha demostrat que els valors menors a -30 kcal/mol indiquen una gran eficàcia del fàrmac contra la diana [5]. Els valors superiors a -10 kcal/mol són considerats massa febles per ser considerats com a possibles fàrmacs. Finalment, els valors positius indiquen que s'ha generat energia arrel de l'enllaç donat que el fàrmac i la diana s'han rebutjat.

Es pot pensar en aquest valor com a la quantitat d'energia que caldria aportar a l'enllaç per tal de trencar-lo (energia = 0): com més negatiu és el valor més energia s'haurà d'aportar i per tant més fort serà l'enllaç. La majoria de casos en què l'energia és positiva es deuen a errors de càlcul en el procediment.

2.2.1.2. Creació de models predictius

Els models estadístics són equacions matemàtiques utilitzades per reproduir fenòmens observats de la forma més exacta possible.

Per altra banda, es parla de models predictius quan aquests models són creats per tal de predir més acuradament la probabilitat d'un resultat. Aquests models són construïts amb dades conegudes (dades d'entrenament), i són aplicats a dades de test. En aquest context, el model construït ens permetria saber com d'acurades són les dades de test respecte a les dades d'entrenament.

Per exemple, donat un conjunt de valors que representen els beneficis d'una empresa al llarg dels últims anys, un model predictiu faria la predicció de quins poden ser els beneficis esperats els pròxims anys. En aquest exemple, les dades d'entrenament serien els beneficis passats, i els valors de test podrien ser un conjunt de valors que el model classificaria com a realistes o no.

En el cas de l'aplicació SEABED, es tractarà de construir models predictius que ens permetin predir l'activitat entre fàrmacs (molècules) i dianes terapèutiques (proteïnes). Per tal d'obtenir aquestes prediccions, els usuaris hauran de crear models predictius utilitzant molècules d'entrenament, per posteriorment aplicar aquests models sobre molècules de test. En altres paraules, els usuaris construiran models amb fàrmacs que se sap quina activitat tenen quan s'apliquen a una diana terapèutica, i, posteriorment, aplicaran aquests models a un nou conjunt de molècules, per tal de comprovar si el comportament d'aquest conjunt de test és similar a l'inicial.

Els resultats de crear un model predictiu es representaran en una corba ROC (Receiver Operating Characteristic, veure Annex 1).

L'aplicació ofereix dues metodologies per crear aquests models: QSAR i PSAR. La principal diferència entre aquests dos mètodes és l'origen de l'activitat de les molècules d'entrenament.

2.2.1.2.1. MODELS QSAR

En els models Quantitative structure–activity relationship (QSAR), l'activitat de les molècules d'entrenament contra una diana determinada és coneguda prèviament, ja que és el resultat d'experiments ja realitzats.

En aquesta metodologia, les variables predictores utilitzades són un conjunt de les propietats físico-químiques de les molècules i de descriptors teòrics bidimensionals. Per tal de crear els models es disposarà de diferents tipus d'algorismes classificadors (o algorismes de *machine learning*): Logit [6], Probit [7], *Linear Discriminant Analysis* [8], *Naive Bayes* [9], Kernel Support Vector Machines [10] , Regularized Discriminant Analysis [11] i Random Forest [12].

Aquests són els que analitzaran les dades de test i determinaran en quin grau es corresponen amb les dades d'entrenament.

Els passos a seguir per tal de crear un model utilitzant aquesta metodologia seran:

1) Calcular els descriptors de les molècules d'entrenament (de les quals es coneix la seva activitat)

2) Crear el model predictiu utilitzant un algorisme de *machine learning* determinat.

2.2.1.2.2. MODELS PSAR

El principal problema de la creació de models predictius utilitzant la metodologia QSAR és que s'ha de disposar de l'activitat prèvia de les molècules d'entrenament. No obstant, això no és així en molts casos. Una via alternativa a aquesta s'anomena PSAR (Pseudo-quantitative structure-activity relationship), en la qual s'utilitzen els resultats de realitzar un *docking* contra la diana per tal d'assignar pseudo-activitat a les molècules d'entrenament. Parlem de pseudo-activitat donat que els resultats del *docking* no reflexen amb tanta precisió l'activitat com si aquests resultats provinguessin d'experiments previs.

Així doncs, els passos per tal de crear un model seguint la metodologia PSAR són:

1) Realitzar un *docking* entre les molècules d'entrenament i la diana terapèutica, i obtenir un rànquing ordenat de les molècules segons la seva activitat

2) Seleccionar les molècules que han obtingut el millor resultat de *docking* (normalment un valor menor que el 10%) i marcar-les com a pseudo-actives i marcar la resta com a pseudo-inactives.

3) Calcular els descriptors de les molècules d'entrenament (a les quals s'ha assignat un valor de pseudo-activitat o pseudo-inactivitat)

4) Crear el model predictiu utilitzant un algorisme de *machine learning* determinat.

D'aquesta manera, es pot dir que la metodologia PSAR equival a realitzar un *docking*, escollir un percentatge reduït com a pseudo-actius i procedir de manera anàloga a la metodologia QSAR. Així doncs, caldrà implementar la metodologia QSAR i el *docking* de manera que aquests es puguin connectar fàcilment per tal d'implementar la metodologia PSAR.

Cal considerar PSAR com a una alternativa a la metodologia QSAR, i el fet d'utilitzar-ne una a una altra dependrà de cada cas en concret. Per exemple, en els casos en què no es disposi d'una estructura tridimensional sol ser més fàcil utilitzar la metodologia QSAR, mentre que si es disposa de resultats de *docking* serà preferible realitzar la metodologia PSAR.

Finalment, cal destacar que actualment no es coneix cap eina que ofereixi la possibilitat de crear models seguint la metodologia PSAR de manera automàtica, fet que aporta molta utilitat a l'aplicació SEABED.

2.2.1.3. Aplicació de models predictius a dades de test

Tal com s'ha explicat en l'apartat anterior, el principal objectiu de crear models predictius és utilitzar-los sobre dades de test per tal de predir el seu comportament respecte a les dades d'entrenament.

Amb aquesta meta l'aplicació permetrà aplicar els models, construïts seguint qualsevol de les metodologies anteriors, a fàrmacs de test per tal de comprovar si aquestes tenen un comportament similar al de les molècules d'entrenament sobre la mateixa diana

terapèutica. Si això és així, es podrà dir que aquests fàrmacs de test generen activitat quan interaccionen amb la diana, i que, per tant, són un fàrmac potencial.

Per tal de realitzar aquest *workflow*, els usuaris només hauran de crear un model predictiu, tal i com s'ha comentat anteriorment, i posteriorment seleccionar quines són les molècules de test. Un cop finalitzat el càlcul, l'usuari obtindrà una llista on apareixeran totes les molècules amb l'activitat predita: actiu o inactiu.

Aquest workflow es realitza amb l'objectiu de reduir grans llibreries de molècules per quedar-se només amb aquelles que tenen activitat contra la diana en qüestió. Aquestes molècules són les que són fàrmacs potencials, i caldrà estudiar-les amb altres metodologies per tal de discernir si finalment tenen potencial per esdevenir fàrmacs reals.

Com a resultat d'aplicar els models es donarà un valor (entre 0 i 1) a cada molècula de test, que representarà la probabilitat que aquesta molècula tingui activitat o no.

3 Especificació

3.1. Anàlisi de requeriments

En aquest apartat es tractarà l'anàlisi dels requeriments de l'aplicació SEABED. L'objectiu principal en l'etapa d'anàlisi de requeriments és el de trobar les necessitats que tenen els clients (o usuaris, en el nostre cas) i que esperen resoldre mitjançant l'eina. A continuació s'estudiaran tant els requeriments funcionals com els no funcionals de l'aplicació.

Per tal de facilitar una millor comprensió dels requeriments, cadascun d'aquests serà descrit amb una plantilla seguint el format:

Requeriment:	Identificador
Nom:	Nom del requeriment
Descripció i justificació:	Breu descripció del requeriment i justificació de la seva correctesa

3.1.1. Requeriments funcionals

Els requeriments funcionals especifiquen les tasques que s'han de realitzar per tal de satisfer funcionalitats requerides pels usuaris.

Requeriment:	RF1
Nom:	Crear models predictius QSAR
Descripció i justificació:	L'usuari serà capaç de crear models predictius fent servir la metodologia QSAR.

	Com a entrada s'utilitzaran dades proporcionades pels usuaris o bé dades provinents de bases de dades accessibles públicament.

Requeriment:	RF2
Nom:	Crear models predictius PSAR
Descripció i justificació:	L'usuari serà capaç de crear models predictius fent servir la metodologia PSAR.

Requeriment:	RF3
Nom:	Aplicar models predictius
Descripció i justificació:	Un cop creats, l'usuari podrà aplicar els seus models predictius a dades de test per tal de comprovar si aquestes poden ser un fàrmac potencial

Requeriment:	RF4
Nom:	Simular *dockings*
Descripció i justificació:	L'usuari serà capaç de crear models predictius fent servir la metodologia PSAR.

	La principal diferència és que mentre amb els models QSAR l'usuari sap per avançat si les molècules tenen activitat o no amb la diana, mentre que amb el mètode PSAR aquesta activitat s'estima a partir d'un docking previ. L'activitat sorgida d'aquesta estimació es denomina pseudo-activitat.

Requeriment:	RF5
Nom:	Descarregar resultats de docking i aplicar models
Descripció i justificació:	L'usuari podrà descarregar-se els resultats dels dockings o d'aplicar models per tal de poder-los utilitzar en altres aplicacions o fer-ne un estudi posterior

Requeriment:	RF6
Nom:	Modificar dades d'entrenament
Descripció i justificació:	L'usuari podrà modificar els fitxers que pugi a l'aplicació, permetent que s'esborrin molècules dels fitxers d'entrenament

Requeriment:	RF7
Nom:	Mostrar les característiques d'un model
Descripció i justificació:	Un cop finalitzada la creació d'un model, l'usuari podrà consultar els paràmetres que va escollir a l'hora de crear-lo

Requeriment:	RF8
Nom:	Notificar els usuaris quan finalizin processos llargs
Descripció i justificació:	Els usuaris podran deixar un mail per a rebre notificacions quan els processos que triguen més temps finalitzen

3.1.2.Requeriments no funcionals

Els requeriments no funcionals defineixen les qualitats que ha de tenir l'aplicació per tal de ser d'alguna utilitat als usuaris.

Requeriment:	RNF1
Nom:	Usabilitat
Descripció i justificació:	L'aplicació ha de ser fàcil de fer servir i intuitiva, ja que que no s'espera que els seus usuaris tinguin un coneixement informàtic elevat.

Requeriment:	RNF2
Nom:	Extensibilitat
Descripció i justificació:	L'aplicació ha d'estar implementada de manera que es puguin anar afegint noves funcionalitats ràpidament per tal d'adaptar-se a demandes futures

Requeriment:	RNF3
Nom:	Portabilitat

Descripció i justificació:	L'aplicació ha de ser fàcilment portable a altres servidors per tal de donar un millor servei als seus usuaris

Requeriment:	RNF3
Nom:	Robustesa
Descripció i justificació:	L'aplicació ha de ser conscient de quan s'ha produït un error en algun dels càlculs i informar l'usuari per tal que actuï al respecte

Requeriment:	RNF4
Nom:	Seguretat
Descripció i justificació:	L'aplicació ha de garantir l'accés segur dels seus usuaris i encriptar les seves dades personals

3.2. Model de casos d'ús

Abans de començar a parlar dels casos d'ús que conformen l'aplicació, caldrà introduir quins seran els *stakeholders*, és a dir, quines seran les persones que donaran ús a l'aplicació.

SEABED és una eina de suport al descobriment de nous fàrmacs, i serà utilitzada per investigadors en el camp de la bioinformàtica. Així doncs, aquests investigadors seran els *stakeholders* de la nostra aplicació.

3.2.1. Tipus d'usuaris

Tal com s'ha comentat anteriorment, l'eina permetrà la creació d'usuaris (anònims o registrats). La diferència principal és que els usuaris registrats proporcionaran dades personals per registrar-se a la web (nom d'usuari, *email*, contrasenya i institució) i els usuaris anònims podran accedir sense registrar cap dada. El principal inconvenient dels usuaris anònims serà que la seva informació s'esborrarà transcorregut un mes des de la seva creació. Per tal d'evitar que això passi, els usuaris anònims podran registrar-se en qualsevol moment sense perdre cap de les seves dades a l'aplicació.

Per altra banda, per tal de donar prioritat als usuaris interns del departament, s'ha creat un altre subconjunt dintre dels usuaris registrats. Aquests usuaris interns només tindran més prioritat i recursos a l'hora de llençar càlculs, però a part d'això seran iguals que els usuaris registrats.

Així doncs, per tal de simplificar, quan es defineixin els actors en els casos d'ús es parlarà només d'usuaris i d'usuaris anònims. Aquesta distinció és necessària per tal de poder referenciar l'únic cas d'ús propi dels usuaris anònims: Registrar usuari. Com que tota la resta de casos d'ús és la mateixa per a usuaris registrats (interns o externs) i usuaris anònims, els englobarem tots en un superconjunt anomenat usuaris.

En el següent diagrama es mostra la jerarquia d'usuaris que hi ha a l'aplicació SEABED:

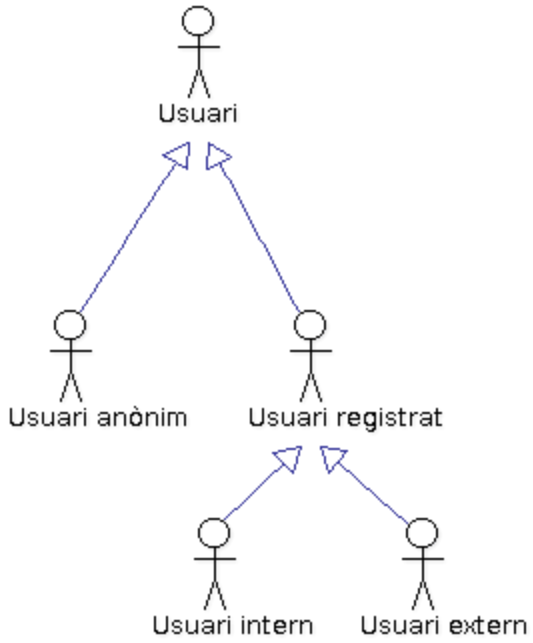


Figura: Jerarquia d'usuaris de l'aplicació

3.2.2. Casos d'ús

3.2.2.1. Cas d'ús: Registrar usuari

Actor principal: Usuari anònim

Precondició: -

Disparador: Un usuari anònim selecciona l'opció "Registrar usuari"

Escenari correcte:

1. L'usuari anònim proporciona un nom d'usuari, una contrasenya, un *email*, i (opcionalment) el nom de la seva institució.

a. Si el nom d'usuari ja existeix o la contrasenya o l'*email* no tenen un format correcte s'avisa a l'usuari de l'error i no es continua.

2. Es crea un nou usuari al sistema amb les dades proporcionades

3. Es fa una còpia de les dades de l'usuari anònim (projectes, resultats, models, etc) al nou usuari

4. S'esborra l'usuari anònim del sistema

3.2.2.2. Cas d'ús: Crear projecte

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació

Disparador: Un usuari selecciona l'opció "Nou projecte"

Escenari correcte:

1. L'usuari proporciona un nom per al nou projecte.

    a. Si el nom del projecte ja existeix es mostra un missatge d'error i no es continua

2. L'usuari pot afegir fitxers al nou projecte que li serviran per posteriorment crear models o llençar dockings

3. S'afegeix un conjunt de molècules per defecte a la carpeta del projecte de l'usuari

3.2.2.3. Cas d'ús: Eliminar projecte

Actor principal: Usuari

Precondició: L'usuari te un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim

Disparador: Un usuari selecciona l'opció "Eliminar projecte"

Escenari correcte:

1. L'usuari selecciona un projecte que vol eliminar.

2. S'avisa l'usuari que el projecte no podrà ser recuperat un cop eliminat

3. S'esborren totes les dades del projecte (models, resultats, etc)

3.2.2.4. Cas d'ús: Llençar docking

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim

Disparador: Un usuari selecciona l'opció "Nou docking"

Escenari correcte:

1. L'usuari selecciona el projecte en el qual vol crear el docking.

2. L'usuari dóna un nom al docking.

    a. Si ja existeix un docking o un model amb el mateix nom, s'afegeix un número al final del nom per tal de fer-lo únic.

3. L'usuari selecciona els fitxers on estan emmagatzemades les molècules que vol incloure en el docking

    a. Si no es selecciona cap fitxer es mostra un missatge d'error i no es continua

4. L'usuari selecciona el nombre total de molècules sobre el total que intervindran en el docking

    a. Si el nombre seleccionat és menor que 0 o més gran que el total de molècules es mostra un error i no es continua

    b. Si el nombre seleccionat està en el rang 1 <= x < màxim s'agafen 'x' molècules aleatòriament

5. L'usuari selecciona el *target* (diana terapèutica) del docking. Si es vol realitzar un docking rígid, l'usuari pot introduïr el codi Uniprot de la proteïna o bé pujar un fitxer

.pdb que representi la proteïna. Si es vol realitzar un docking elàstic, l'usuari haurà de pujar dos fitxers, amb l'estructura de la proteïna de referència i la trajectòria d'aquesta

a. Si es selecciona el docking elàstic, es calcularan els *snapshots* que es faran servir posteriorment per al *docking*

6. L'usuari seleccionarà el *pocket* de la proteïna, bé pujant un fitxer amb la representació d'aquest, seleccionant-lo manualment sobre la cadena o bé escollint una opció que el seleccionarà automàticament

7. L'usuari llençarà el docking

a. Si l'usuari ha introduït el seu *email* en el sistema se'l notificarà quan acabi el procés

8. Quan acabi el procés, el resultat s'emmagatzemarà al sistema per poder ser consultat en qualsevol moment

3.2.2.5. Cas d'ús: Crear nou model amb la metodologia QSAR

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim

Disparador: Un usuari selecciona l'opció "Nou model QSAR"

Escenari correcte:

1. L'usuari selecciona el projecte en el qual vol crear el model QSAR.

2. L'usuari dóna un nom al model.

a. Si ja existeix un docking o un model amb el mateix nom, s'afegeix un número al final del nom per tal de fer-lo únic.

3. L'usuari selecciona els fitxers on estan emmagatzemades les molècules d'entrenament

a. Si no es selecciona cap fitxer es mostra un missatge d'error i no es continua

4. L'usuari selecciona els paràmetres per calcular els descriptors

a. Si algun dels paràmetres escollits no està en el format correcte, es mostra un missatge d'error i no es continua

5. L'usuari pot esborrar les molècules que no li interessin abans de crear el model

6. L'usuari seleccionarà quin algorisme de *machine learning* i els paràmetres d'aquest que vol fer servir per crear el model

7. Es crearà el model a partir de totes les dades proporcionades

a. Si l'usuari ha introduït el seu *email* en el sistema se'l notificarà quan acabi el procés

8. Quan acabi el procés, el resultat s'emmagatzemarà al sistema per poder ser consultat en qualsevol moment

3.2.2.6. Cas d'ús: Crear nou model amb la metodologia PSAR

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim

Disparador: Un usuari selecciona l'opció "Nou model PSAR"

Escenari correcte:

1. L'usuari selecciona el projecte en el qual vol crear el model PSAR.

2. L'usuari dóna un nom al model.

a. Si ja existeix un docking o un model amb el mateix nom, s'afegeix un número al final del nom per tal de fer-lo únic.

3.2.2.7. Cas d'ús: Mostrar informació de model

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un model com a mínim

Disparador: Un usuari selecciona l'opció "Mostrar informació de model"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el model que vol consultar.
2. L'usuari selecciona el model que vol consultar.
3. Es mostra la informació del model seleccionat: paràmetres utitlitzats en la creació i corba ROC que representa el resultat

3.2.2.8. Cas d'ús: Eliminar model

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un model finalitzat com a mínim

Disparador: Un usuari selecciona l'opció "Eliminar model"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el model que vol eliminar.
2. L'usuari selecciona el model que vol eliminar.
3. S'avisa l'usuari que el model no podrà ser recuperat un cop eliminat
4. S'esborren les dades del model

3.2.2.9. Cas d'ús: Eliminar resultat

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un resultat com a mínim

Disparador: Un usuari selecciona l'opció "Eliminar resultat"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el resultat que vol eliminar.
2. L'usuari selecciona el resultat que vol eliminar.
3. S'avisa l'usuari que el resultat no podrà ser recuperat un cop eliminat
4. S'esborren les dades del resultat

3.2.2.10. Cas d'ús: Mostrar resultat

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un resultat com a mínim

Disparador: Un usuari selecciona l'opció "Mostrar resultat"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el resultat que vol consultar.
2. L'usuari selecciona el resultat que vol consultar.
3. Es mostra el resultat seleccionat (pot ser el resultat d'un *docking* o el resultat d'aplicar un model)

3.2.2.11. Cas d'ús: Mostrar informació de model

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un model finalitzat com a mínim

Disparador: Un usuari selecciona l'opció "Informació del model"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el model que vol consultar.
2. L'usuari selecciona el model que vol consultar.
3. Es mostra la informació

3.2.2.12. Cas d'ús: Afegir fitxer d'entrenament

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim

Disparador: Un usuari selecciona l'opció "Afegir fitxer d'entrenament"

Escenari correcte:

1. L'usuari selecciona el projecte al que vol afegir el fitxer d'entrenament.
2. L'usuari selecciona l'opció "Afegir fitxer d'entrenament"
3. L'usuari selecciona el fitxer que conté les dades d'entrenament
    a. L'usuari pot marcar les molècules del fitxer d'entrenament com a actives o inactives
4. El fitxer d'entrenament és afegit al projecte seleccionat prèviament

3.2.2.13. Cas d'ús: Modificar fitxer d'entrenament

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un fitxer on hi ha dades de les molècules d'entrenament

Disparador: Un usuari selecciona l'opció "Modificar fitxer d'entrenament"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el fitxer que vol modificar.
2. L'usuari selecciona el fitxer que vol modificar.
3. Es mostra un llistat amb les molècules incloses al fitxer seleccionat
4. L'usuari elimina les molècules que no li interessen del fitxer amb un botó d'"Eliminar molècula" situat al costat de cada molècula
5. L'usuari selecciona l'opció "Finalitzar quan ha acabat d'eliminar molècules
6. Es modifica el fitxer d'entrenament eliminant les molècules seleccionades

3.2.2.14. Cas d'ús: Eliminar fitxer d'entrenament

Actor principal: Usuari

Precondició: L'usuari té un compte (anònim o registrat) a l'aplicació i disposa d'un projecte com a mínim amb un fitxer d'entrenament com a mínim

Disparador: Un usuari selecciona l'opció "Eliminar fitxer d'entrenament"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el fitxer d'entrenament que vol eliminar.
2. L'usuari selecciona el fitxer d'entrenament que vol eliminar.
3. S'avisa l'usuari que el fitxer d'entrenament no podrà ser recuperat un cop eliminat

4. S'esborra el fitxer d'entrenament del sistema

3.2.2.15. Cas d'ús: Notificar usuari

Actor principal: Aplicació

Precondició: L'usuari té un projecte amb un workflow on ha llençat un procés llarg i ha emmagatzemat un *email* per rebre notificacions de treballs

Disparador: El procés llarg ha finalitzat

Escenari correcte:

1. Un usuari vol executar un procés llarg (calcular descriptors, simular docking o crear model) en algun dels sistemes de cues
2. Es detecta que ha finalitzat el procés
    a. Si l'usuari ha emmagatzemat un *email* per rebre notificacions, se li envia un correu per informar-lo que el procés ha acabat

3.2.2.16. Cas d'ús: Continuar workflow

Actor principal: Usuari

Precondició: L'usuari té un projecte que conté un workflow no finalitzat

Disparador: Un usuari selecciona l'opció "Continuar *workflow*"

Escenari correcte:

1. L'usuari selecciona el projecte que conté el workflow que vol continuar.
2. L'usuari selecciona el workflow que vol continuar.
3. L'usuari continua amb el *workflow* des del punt on va quedar pendent

3.2.3. Diagrama de casos d'ús

Figura: Diagrama de casos d'ús de l'aplicació

4. Anàlisi de l'aplicació

En el següent apartat s'analitzaran els diferents aspectes que conformen l'aplicació SEABED. Primerament, es parlarà dels sistemes de cues utilitzats per gestionar l'execució dels processos, posteriorment es detallarà l'estructura de l'aplicació i els elements que la formen. Per altra banda, es parlarà del hardware sobre el qual s'executa l'aplicació i finalment es mostrarà el model de dades utilitzat per l'aplicació.

4.1. Sistemes de cues

Els sistemes de cues s'utilitzen per a gestionar l'execució de múltiples processos en una o més màquines. Els processos s'executen mentre queden recursos disponibles, i quan aquests s'esgoten s'esperen fins que en tornen a haver disponibles en el que s'anomena un cua de processos. Els sistemes gestors de cues són els encarregats de manegar aquesta cua. A més a més, solen tenir diferents polítiques d'actuació, per tal de poder gestionar l'execució dels processos d'una manera més eficient. En aquestes polítiques d'actuació es tracta com entraran els processos a les cues, els tipus de prioritats entre dos processos, el temps d'execució màxim d'un procés, etc.

En l'aplicació SEABED s'han utilitzat dos gestors de cues diferents, donat que es fan servir dos tipus de màquines diferents segons els workflows que es vulguin realitzar: els dockings es realitzen en un cluster de càlcul del departament de Life Sciences, i tots els altres càlculs (creació i aplicació de models, càlcul de descriptors i *snapshots*, etc) s'executen a les estacions de treball del propi grup. Això és degut a que els *dockings* són (amb diferència) els càlculs més pesats, i per tant requereixen d'una potència de càlcul més elevada per ser executats en un temps raonable. La diferència de temps entre els processos executats va de dies en el cas dels *dockings,* a minuts o poques hores en els altres càlculs, sempre depenent de la mida de l'entrada.

A continuació es donarà una breu descripció dels dos sistemes de cues utilitzats.

4.1.1. Oracle Grid Engine

El gestor de cues utilitzat en el servidor principal s'anomena Oracle Grid Engine [13]. Aquest sistema, desenvolupat per Sun Technologies, serveix per distribuir l'execució de

treballs entre diferents màquines que pertanyen a una mateixa *granja* de servidors o bé en un *cluster* d'un supercomputador. En el nostre cas, es faran servir les estacions de treball dels diferents membres del grup en el qual s'ha desenvolupat el projecte.

El principal avantatge que ofereix aquest sistema és que ofereix moltes opcions de configuració diferents: crear diferents cues amb diferents propietats cadascuna, configurar un nombre màxim de recursos a utilitzar en cada màquina, etc. Això permet que es pugui configurar cada màquina de manera que se'n pugui treure el màxim rendiment, independentment del hardware i de la càrrega de treball a la que la sotmet el propi usuari.

Aquest gestor ja era accedit per altres aplicacions desenvolupades prèviament, per tant la seva integració amb *SEABED* no va presentar molts problemes.

4.1.2. SLURM Queue Manager

SLURM (Simple Linux Utility for Resorce Manager [14]) és un gestor de recursos lliure dissenyat per treballar amb *clusters* de màquines Linux de diferent mida. Aquest és el sistema de cues utilitzat per realitzar càlculs més costosos al cluster de Life Sciences (explicat amb més detall en l'apartat 4.3.3).

Les tres virtuds de SLURM són:

- Permetre accés exclusiu o no exclusiu als usuaris durant un temps determinat per realitzar els seus càlculs.

- Proporciona un framework per tal de poder iniciar, executar i monitoritzar treballs en un conjunt de màquines.

- Mantenir una cua de treballs pendents per tal de gestionar més eficientment els recursos.

Aquest sistema està integrat actualment en 5 dels 10 primers de la llista Top500[7] el mes de Juny de 2013, on es manté un rànquing dels supercomputadors amb més capacitat de càlcul del món. A més a més, també s'utilitza al Mare Nostrum, el supercomputador gestionat pel BSC.

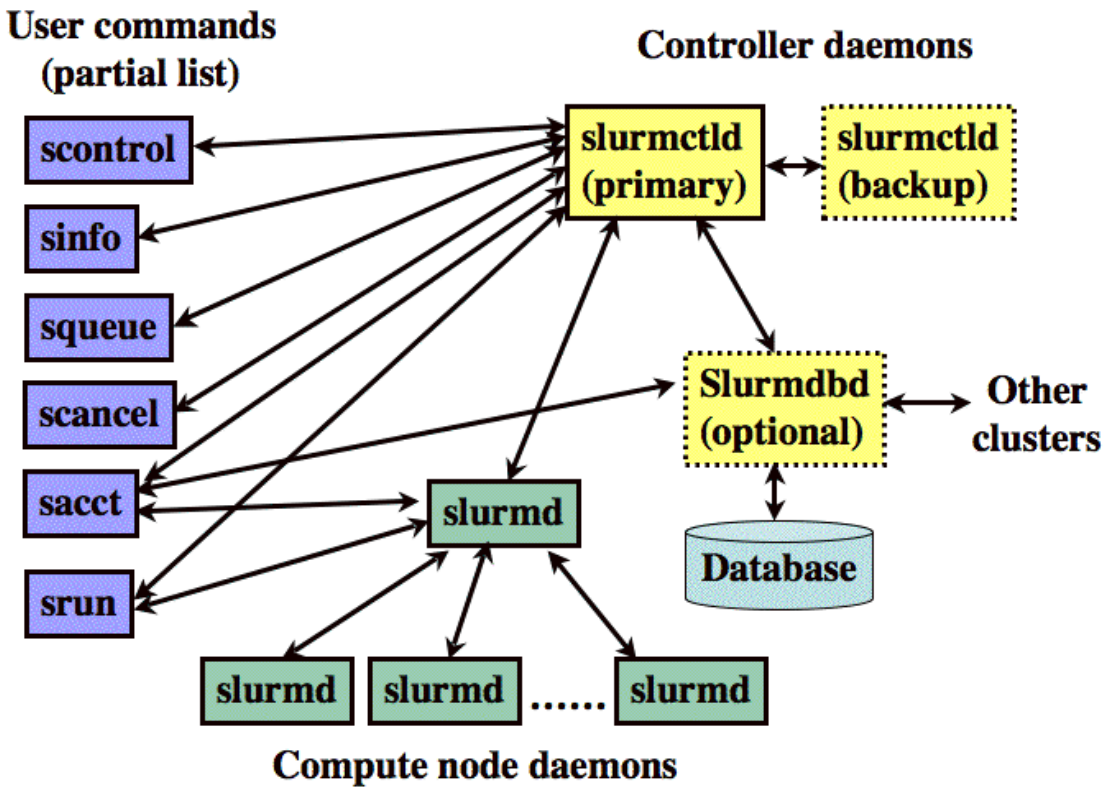


Figura: Diagrama dels components de SLURM

En la figura de dalt es mostren els diferents components que conformen el gestor SLURM. Per una banda, hi ha les comandes que poden executar els usuaris per tal de gestionar la cua: inserció, eliminació, monitorització, etc. Per altra banda, hi ha un controlador central, *slurmctld*, (que pot estar replicat per incrementar la tolerància a fallades), que monitoritza i gestiona tot el volum de càlcul. A més, cadascun dels servidors del *cluster* disposen d'un *daemon*, encarregat de rebre treballs del *slurmctld*, executar-lo i retornar els resultats. Per altra banda, SLURM disposa d'una base de dades, on es pot guardar informació persistent sobre l'estat dels treballs, encara que aquesta funcionalitat no es fa servir en la nostra aplicació, ja que es disposa d'un sistema propi que realitza la mateixa funció. Es parlarà d'aquest sistema en el següent apartat.

4.1.3. Gestió dels sistemes de cues

Encara que es tracti de dos sistemes diferents, el funcionament bàsic dels dos gestors de cues explicats anteriorment per tal d'executar un procés qualsevol es podria simplificar en:

1. Moure totes les dades necessàries per executar el procés des del directori de treball a una carpeta accessible pel sistema de cues (directori temporal).

2. Crear un fitxer de configuració del treball, on s'especificaran:

   - Els paràmetres de la cua (cua sobre la qual s'executara el treball, nombre de *cpus* que utilitzarà el procés, directoris pels fitxers de sortida i d'error, credencials de l'usuari de la cua, etc)

   - Les llibreries i variables d'entorn necessàries per executar el procés

   - La comanda que s'encarrega d'executar el procés

3. Encuar el procés utilitzant el fitxer creat

4. Esperar a que finalitzi l'execució del procés

5. Un cop finalitzat el procés, comprovar que no hi ha hagut cap error i moure els resultats des del directori temporal cap al directori de treball

Per altra banda, les principals diferències entre la gestió dels dos sistemes són:

- Les crides utilitzades per encuar, monitoritzar i finalitzar processos són diferents

- El *Queue Daemon* i el sistema de cues del servidor s'executen en la mateixa màquina, mentre que el sistema de cues de Life s'executa en una màquina remota. Això afegeix la complexitat de sincronitzar dades entre les dues màquines, realitzar crides remotes en comptes de crides locals, etc.

- Els formats i els paràmetres dels fitxers dels *jobs* a executar són diferents

4.2. Estructura de l'aplicació

El codi de l'aplicació SEABED es pot dividir en dos components ben diferenciats: per una banda, hi ha les classes que contenen tota la lògica de l'aplicació i la gestió de la seva interfície (component SEABED), i per altra banda hi ha un altre conjunt de classes que serveixen per comunicar l'aplicació amb el sistema de cues (component *QueueSystem*).

Tot i que funcionen conjuntament en l'aplicació, els dos components comentats són majoritàriament independents l'un de l'altre. Les úniques dependències que tenen resideixen en la manera com es connecten. Això afavoreix que es puguin afegir millores en el funcionament intern de qualsevol dels dos components, mentre no es modifiquin els comportaments que gestionen la seva connexió. A continuació es detallaran quins són els elements que conformen cadascun dels components de l'aplicació.

4.2.1. Component SEABED

En aquest és on s'agrupa la lògica pròpia de l'aplicació: gestió d'usuaris i de la interfície, *wrappers* de les llibreries que realitzen els càlculs dels workflows i de les taules de la base de dades, organització del sistema de fitxers, etc. És el component que representa el nucli de l'aplicació.

4.2.1.1. Gestió d'usuaris

La informació dels usuaris ve representada directament pel sistema de fitxers. L'aplicació disposa d'una carpeta anomenada *sessions* on es crea una carpeta per a cadascun dels seus usuaris. Dintre de cada carpeta, hi ha un fitxer amb la informació encriptada de l'usuari i tantes carpetes com projectes hagi creat. Dintre el directori de cada projecte es disposarà de diferents carpetes:

- *Models*, on es guarda la informació dels models
- *Results,* on es troben tots els resultats obtinguts en aquell projecte
- *Temp,* que es fa servir com a carpeta per a tractar fitxers temporals
- *Sandbox,* on es troben les dades dels models i els dockings quan els workflows no han acabat
- *Img,* on s'emmagatzemen les imatges de les molècules mostrades a les taules

Per gestionar aquest sistema de fitxers, s'han implementat dues classes: *User* i *FileManager*. La primera es fa servir per obtenir informació de l'usuari: nom del projecte en el que està treballant, dades personals, etc. Pel que fa al *FileManager*, s'encarrega de gestionar els fitxers associats a cadascun dels usuaris: agrupa les seves dades en les carpetes comentades anteriorment, recupera la informació en el fitxer personal dels usuaris, etc.

4.2.1.2. Interfície gràfica de l'aplicació

Un dels components més importants de les aplicacions web és la seva interfície, ja que aquesta té una gran influència en la valoració global de l'eina per part dels seus usuaris.

En el cas de SEABED, la interfície gràfica s'ha construït utilitzant components del framework *Vaadin*. Aquests s'han anat agrupant per a formar les vistes que es mostren als usuaris i que els hi permeten interaccionar amb les dades i realitzar els càlculs que desitgin.

La zona de treball de l'aplicació és un panell al qual se li poden anar afegint pestanyes per tal de poder treballar en dos processos paral·lelament. El funcionament és molt similar al d'un navegador web, en el que es disposa de múltiples pestanyes per accedir a diferents llocs simultàniament. En aquest panell és on els usuaris podran treballar en els seus *workflows*, i per tant és on es concentraran la majoria dels components.

Per facilitar l'ús de l'eina, en els *workflows* s'agruparan els elements de la interfície a utilitzar en cada pas en un panell desplegable verticalment, de manera que els usuaris podran recuperar la informació d'un pas anterior en qualsevol moment.

Pel que fa al codi, els elements de la interfície de cadascuna de les vistes s'han agrupat en classes. Cadascuna d'aquestes classes també gestiona els events que es produeixen a través de la vista (polsar botons, marcar/desmarcar caselles, etc). S'han creat dues classes més genèriques per als dos panells globals comentats anteriorment. Aquests panells serveixen de contenidors de les altres vistes més específiques.

4.2.1.3. Wrappers de les llibreries bioinformàtiques

Per tal de realitzar els càlculs involucrats en els workflows de l'aplicació, s'utilitzen unes llibreries ja desenvolupades abans de l'inici del projecte. Aquestes llibreries tenien l'inconvenient que eren només accessibles a través de la línia de comandes d'una terminal, i a més tenien múltiples dependències amb altre *software* que calia ser instal·lat si es volien utilitzar.

Un dels objectius principals del projecte és facilitar l'accés a aquestes llibreries a través d'una eina web. Per tal d'abstreure les particularitats d'aquestes llibreries s'han creat classes *wrapper* per a cadascuna d'elles. Aquestes classes serveixen per embolcallar la informació de cadascuna de les crides de la línia de comandes en objectes que poden ser tractats per la resta de l'aplicació.

Cadascuna d'aquestes classes emmagatzema els paràmetres necessaris per a treballar amb la llibreria a la que representa, i ofereix mètodes per poder modificar-los o obtenir-los per treballar amb ells. A més a més, també tenen un mètode que retorna quina és la signatura de la comanda a utilitzar. Aquesta comanda és la que s'invoca des de l'aplicació en fils independents per tal de llençar els càlculs involucrats al sistema de cues que pertoqui.

4.2.1.4. Wrappers de les taules de la base de dades

Com es veurà en l'apartat 4.3."Model de dades", s'ha dissenyat una base de dades relacional per tal de gestionar i treure profit de les dades de l'aplicació. En aquell apartat s'analitzarà amb més profunditat el model de dades de l'aplicació, en aquest només es donarà una breu introducció a l'element que permet gestionar el model de dades de l'aplicació.

Amb l'objectiu de treure profit de la base de dades creada per a l'aplicació, s'han implementat un conjunt de classes que fan referència a cadascuna de les taules de la base de dades. Aquestes classes tindran la funció d'obtenir informació de la taula a la que referencien, a més de poder crear, modificar i eliminar els registres presents a la taula.

El principal benefici d'haver desenvolupat aquestes classes és que es podrà treballar amb objectes Java, i per tant serà molt més senzill comunicar la base de dades amb l'aplicació.

4.2.2. Component Queue System

Per tal d'interaccionar amb els dos gestors de cues, s'ha utilitzat un sistema implementat prèviament per a un altre projecte del grup. D'aquí en endavant, anomerarem a aquest sistema *Queue System*. Els components del *Queue System* són: *Queue Manager, Queue Daemon, Abstract Job, Abstract Queue* i una base de dades.

La classe *Abstract Job* referencia els treballs que s'executen a les cues. Així doncs, cada tipus de treball que es llença a les cues implementarà les funcions abstractes d'aquesta classe. Aquestes funcions implementaran les funcionalitats comuns en tots els treballs: crear els fitxers que s'utilitzaran per encuar els treballs, llençar els treballs a les cues i realitzar el post-processament dels treballs (moure els fitxers de resultats als directoris dels usuaris, enviar els correus de notificació als usuaris, netejar els fitxers temporals, etc).

Per altra banda, l'*Abstract Queue* és una classe abstracta que serveix per representar els gestors de cues. En el desenvolupament de l'aplicació s'ha aprofitat la classe *SGEQueue* que treballa amb les cues d'Oracle del servidor, i s'ha implementat una nova classe, *LifeClusterQueue,* que gestiona la cua del *cluster* de Life Science. Aquestes classes tenen mètodes que permeten obtenir informació de les cues: llistat de treballs, estat de la cua, nombre de treballs màxim de la cua, etc.

El *Queue Manager* és l'encarregat d'executar les comandes utilitzades per encuar els processos, i d'inserir-los a la base de dades un cop han estat llençats. A més a més, també pot aconseguir l'estat dels treballs a través de les classes que implementen l'*Abstract Queue*.

Finalment, hi ha el *Queue Daemon,* un procés independent a l'estat del servidor de l'aplicació que va comprovant quins treballs estan pendents de finalitzar a la base de dades. Un cop finalitzen, el *Queue Manager* obtindrà la seva informació a través de les cues (*SGEQueue i LifeClusterQueue*). Paral·lelament, el *Queue Daemon* s'encarrega

d'actualitzar la base de dades quan canvia l'estat dels treballs, i és crida el post processament dels treballs quan aquests acaben.

Es fa servir una base de dades per tal de mantenir un registre persistent dels treballs que s'executen en les dues cues. D'aquesta manera, si el *Queue Daemon* s'atura, es podrà recuperar l'estat dels treballs que s'estaven executant en el moment previ a l'aturada. A la base de dades s'hi guarden les característiques de tots els treballs que es llencen a les cues:

- id: Identificador únic del treball

- idQueue: Identificador de la cua sobre la qual s'ha llençat el treball

- type: Tipus de treball: docking, càlcul de descriptors, nou model...

- state: Estat del treball: encuat, en execució o finalitzat

- error: Codi per indicar si un treball ha finalitzat amb error (1) o s'ha executat correctament (0)

- localFilesFolder: carpeta on hi ha tots els arxius necessaris per executar un procés.

- clusterFilesFolder: carpeta temporal on es copien tots els arxius del localFilesFolder i es creen els fitxers necessaris per al sistema de cues (fitxers de configuració de les cues, fitxers de sortida o error, etc)

- estimatedSize: mida estimada de tots els arxius temporals emmagatzemats a clusterFilesFolder

- cluster: Identificador del tipus de màquina on s'executaran els treballs (SGE o LifeScience)

- queueStart: instant de temps en el que el procés s'encua

- exeStart: instant de temps en el que el procés comença la seva execució

- exeEnd: instant de temps en el que el procés finalitza la seva execució

- objIds: array en el que s'emmagatzema informació que cal recuperar quan finalitzen els processos: localFilesFolder, *email* de notificació, etc.

4.1.3.2. Funcionament

La interacció entre l'aplicació SEABED i el *Queue System* queda definida en els següents passos:

1. Un usuari comença un dels workflows que permet realitzar l'aplicació. Qualsevol dels *workflows* necessita llençar un càlcul (més o menys costós) en un dels dos sistemes de cues comentats anteriorment (SGE o SLURM).

2. Quan s'arriba al punt d'executar un càlcul en un dels dos sistemes de cues, es crea un fil d'execució independent i es mostra a l'usuari un diàleg de progrés informant-lo de que el seu procés s'està executant. En el nou fil creat, s'executarà l'*Abstract job* que referencia el procés necessari. Per altra banda, l'usuari podrà continuar navegant per l'aplicació donat que el càlcul és independent del funcionament de l'aplicació gràcies a la utilització del nou fil d'execució.

3. L*'Abstract Job* executat s'encarrega de crear el fitxer amb la configuració del treball (configuració de variables, comandes a executar, límit de temps d'execució del treball, etc). El *Queue Manager* encua el procés i afegeix una nova entrada a la base de dades amb la informació del treball (tipus de treball, cua a la qual s'ha enviat, identificador del treball en aquella cua, etc). El camp d'estat del treball s'inicilialitza a encuat.

4. Periòdicament, el *Queue Daemon* obté els treballs que encara no estan marcats com a finalitzats a la base de dades, i demana a la cua pertinent (*SGEQueue* o *LifeClusterQueue*) si han canviat d'estat. En cas que hagin passat d'estar encuats a executant-se, només es canvia el camp d'estat i la data d'inici de l'execució a la base de dades.

5. Quan el procés finalitza, el *Queue Daemon* crida la funció que post-processa els treballs per tal de moure els fitxers de resultat a la carpeta pertinent de l'aplicació, i marca el procés com a finalitzat a la base de dades i li assigna una data de finalització.

6. Finalment, el fil que executa l'Abstract Job notifica a l'usuari (mitjançant un correu electrònic) que el seu càlcul ja ha finalitzat i que per tant pot continuar amb el seu *workflow*.

a. Si l’usuari havia sortit de l’aplicació, podrà continuar a partir del següent pas a través d’un enllaç “Next” del workflow que estava creant.

b. Si l’usuari no havia sortit de l’aplicació, es substituirà el diàleg de progrés de l’aplicació per un botó “Next” que el permetrà continuar al següent pas.

4.3. Hardware

En aquest apartat es parlarà del hardware que s'utilitza per a què l'aplicació SEABED funcioni correctament, és a dir, veurem quin tipus de màquines s'utilitzen i què s'hi realitza en cadascun d'ells. Dividirem els tipus de màquines en tres conjunts: un servidor principal que s'encarrega de gestionar l'aplicació, i dos *clusters*: un intern al grup i l'altre extern. Aquests clusters es faran servir per realitzar els càlculs que es necessiten per als *workflows*[1].

4.3.1. Servidor principal

Anomenem servidor principal a la màquina encarregada de gestionar l'execució de l'aplicació. A més, aquesta màquina s'utilitza per tots els membres del grup com a servidor central: s'hi emmagatzema tot el *software* desenvolupat al grup (juntament amb les llibreries necessàries per a que funcioni), bases de dades, el repositori del *Git* utilitzat per mantenir les diferents versions de codi i dades resultants dels experiments realitzats.

Tal com s'ha comentat anteriorment, s'utilitza Tomcat[15] com a servidor de l'aplicació. Tomcat és un servidor d'aplicacions Java i contenidor de *servlets* (objectes de Java que corren dintre d'un servidor i n'extenen les seves funcionalitats). Aquesta tecnologia ha estat escollida donat que és un dels requeriments del framework *Vaadin* [16], tot i que actualment és utilitzat en moltes aplicacions web amb un volum de trànsit molt elevat (per exemple LinkedIn, Paypal o eBay [17]). La seva popularitat ha fet que fos possible trobar solucions a problemes comuns que s'han anat trobant al llarg del desenvolupament de l'aplicació.

Per altra banda, en aquesta màquina també s'hi troba el servidor de bases de dades on s'ha creat la nova base de dades que en un futur donarà suport a l'aplicació. S'utiliza MySQL[18] com a sistema de gestió de la base de dades (Relational Database Management System (RDMS)). S'ha escollit aquest RDMS donat que ja existien aplicacions dintre del grup que el feien servir dintre el mateix servidor, i per tant seria més fàcil aprofitar part del treball ja realitzat en aquestes aplicacions.

En aquest servidor també s'hi executa el *daemon* del *Queue System*, que, com s'ha comentat anteriorment, és l'encarregat de gestionar l'estat de les execucions dels treballs

a les cues. Finalment, també s'han assignat dos dels vuit *cores* d'aquest servidor al sistema de cues de l'aplicació, per tant sobre aquest servidor també s'executen els treballs que requereixen menys capacitat de càlcul dels *workflows.*

4.3.2. Estacions de treball

Anomenem estacions de treball als ordinadors d'escriptori dels membres del grup on s'ha desenvolupat l'aplicació SEABED. Aquests ordinadors són els que estan associats al sistema de cues *Oracle Grid Engine.* Es fan servir per realitzar els processos que requereixen menys capacitat de càlcul de l'aplicació: càlcul de descriptors, creació i aplicació de models i selecció de *snapshots*.

Donat que aquests ordinadors són utilitzats pels membres del grup per a treballar diàriament, no s'assignen tots els seus recursos al sistema de cues. Així doncs, s'especifica que com a màxim poden fer servir la meitat dels cores dels que disposen per tal de realitzar els càlculs de l'aplicació. A més, el sistema de cues sap quina és la càrrega de les estacions de treball, i intenta mantenir totes les màquines amb el mateix nivell de càrrega. Amb aquestes mesures s'aconsegueix que els usuaris no notin cap descens en el rendiment i a més es paral·lelitza la càrrega de treball, fent que el temps de càlcul requerit sigui menor. A més a més, aquesta arquitectura escala molt bé horitzontalment, és a dir, si s'anessin afegint més màquines al gestor de la cua es podria donar millor servei als usuaris de l'aplicació.

4.3.3. Life Sciences Cluster

Els membres del grup de *Life Sciences* del BSC disposen d'un *cluster* de càlcul molt més potent que les *workstations* per executar els càlculs més complexos. D'aquesta manera s'aconsegueix que el temps invertit per a realitzar els experiments es vegi reduït dràsticament. El cluster disposa d'un total de 252 *cores* i 1376GB de memòria RAM.

L'aplicació SEABED utilitza una part d'aquest *cluster* de càlcul (concretament 64 *cores* i 256GB de memòria RAM) per tal d'executar els *dockings*, que representen el càlcul més pesat executat des de l'aplicació.

Com s'ha comentat anteriorment, a l'aplicació hi ha dos tipus d'usuaris registrats: els interns i els externs. Els treballs dels usuaris interns utilitzen 8 *cores* cadascun, mentre que els dels usuaris externs només n'utilitzen 1. Això fa que els treballs dels usuaris interns s'executin amb un temps molt menor que si els executessin usuaris externs. Amb aquesta política s'aconsegueix donar servei a més usuaris al mateix temps, al preu de que els seus treballs triguin més a executar-se, mentre es garanteix que els usuaris interns tinguin un tracte privilegiat.

En la figura següent es mostra un esquema de les màquines utilitzades en l'aplicació SEABED i els components que es fan servir en cadascuna d'elles.

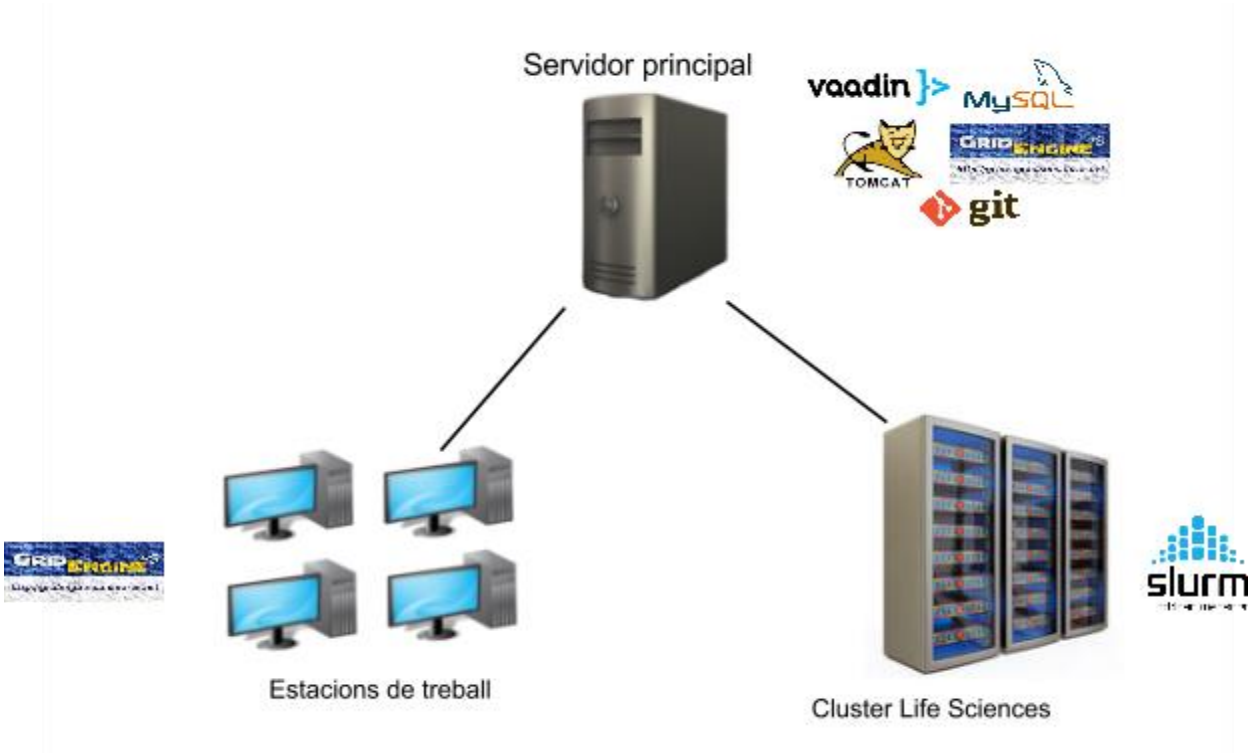


Diagrama: Arquitectura hardware de l'aplicació SEABED

4.4. Model de dades

Des d'un principi, l'aplicació SEABED treballa amb dades emmagatzemades en fitxers. Aquest fet comporta una sèrie d'avantatges i inconvenients, que es podrien resumir en:

Avantatges:

- Els fitxers tenen un format estàndard, fet servir per la multitud de llibreries externes que requereix l'aplicació

- És més fàcil reutilitzar fitxers amb dades precalculades

- Es poden aprofitar dades provinents d'altres aplicacions sempre que aquestes mantinguin el format estàndard

Inconvenients:

- Les dades poden ocupar molt espai (fins a *GB*), per tant és pot perdre molt temps transferint-les

- La mineria de dades és molt més costosa que si es disposés d'una base de dades

- Els usuaris només poden utilitzar fitxers propis

Amb l'objectiu de tractar les dades generades des de l'aplicació per tal de després poder agilitzar la mineria de dades sobre aquestes, en una segona versió de l'aplicació va sorgir la necessitat de dissenyar i implementar un model de dades. A més a més, aquest sistema també hauria de permetre emmagatzemar resultats d'experiments anteriors, per tal de poder ser accedits més fàcilment des de l'aplicació.

En aquest apartat es mostrarà el disseny d'aquest model de dades en detall, a més de la posterior transició a una base de dades MySQL. Aquesta base de dades, si en un futur és integrada amb l'aplicació, serà la que aporti els beneficis comentats anteriorment.

4.4.1. Esquema conceptual

En la figura següent es mostra un esquema conceptual del model de dades que s'utilitza a l'aplicació SEABED. Aquest esquema ha estat el punt de partida per al posterior disseny de la base de dades.

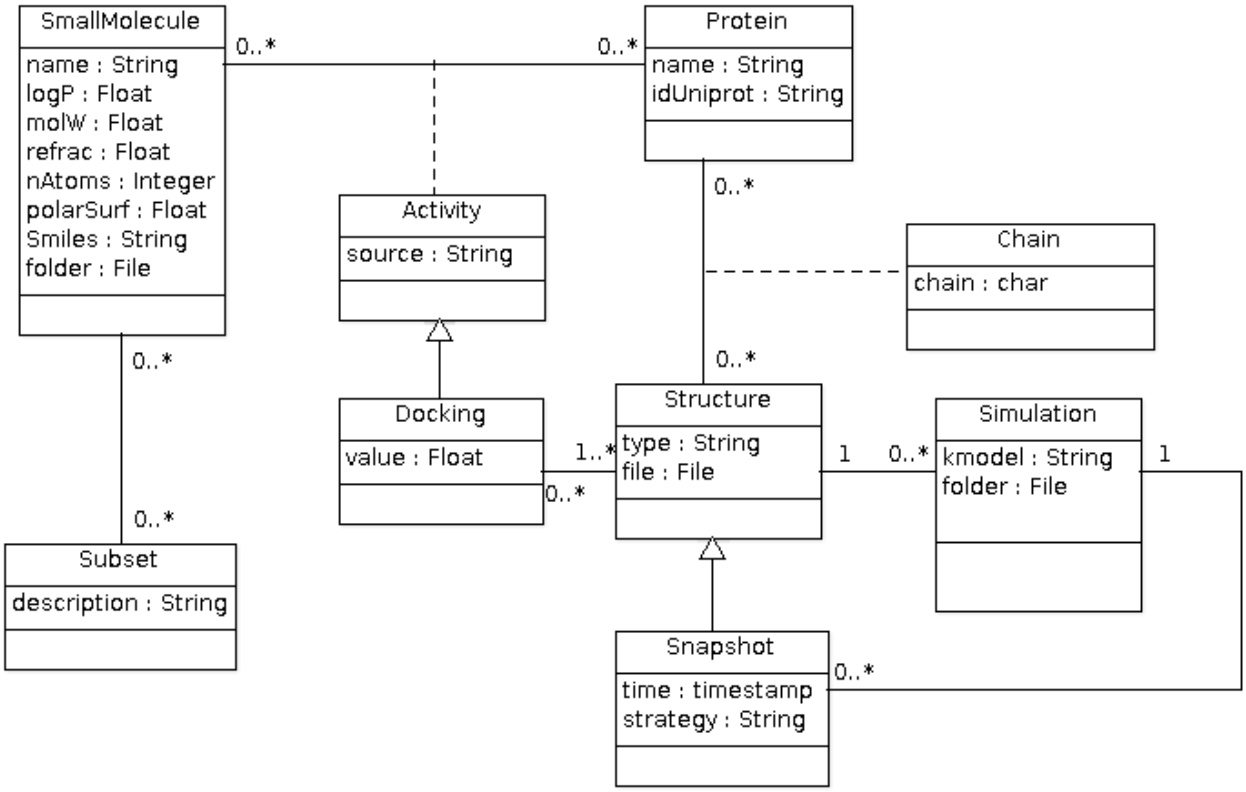


Figura: Model conceptual de dades de l'aplicació SEABED

A continuació s'explicaran els conceptes que apareixen en el diagrama en més detall, així com les relacions que hi ha entre ells. S'utilitzarà la notació (*camp*) en les explicacions per a fer referència al nom que s'ha donat en el diagrama a aquell concepte.

4.4.1.1. Protein

Teòricament, les proteïnes són molècules formades per cadenes d'aminoàcids. Les proteïnes també queden identificades per un nom (*name*).

Actualment hi ha disponibles bases de dades d'accés públic en les quals s'emmagatzema informació d'aquestes proteïnes. En el cas de la nostra aplicació, s'han

utilitzat proteïnes provinents de la base de dades Uniprot[19]. En aquesta aplicació s'utilitzen identificadors propis (*idUniprot*) per mantenir el catàleg de proteïnes.

4.4.1.2. Simulation

En el cas dels *dockings* elàstics, s'intentava aconseguir enllaços més forts intentant modificar fer els *dockings* amb instàncies modificades de la proteïna. Aquestes instàncies s'aconseguien a partir de la estructura base (*Structure*) i una trajectòria. Aquesta trajectòria pot venir referenciada per fitxers en disc (*folder*) o bé estar emmagatzemada en una base de dades externa (*kmodel*).

4.4.1.3. Structure

La *protein* és un concepte més teòric, mentre que la *structure* és la instància física de la *protein.* Fa referència a les diferents disposicions que pot adoptar l'esquelet proteic en l'espai. Aquesta instància és la que es pot cristalitzar en un laboratori per ser utilitzada en un experiment posteriorment. Donat que la qualitat de les cristalitzacions pot variar, poden haver múltiples *structures* diferents entre si però que fan referència a la mateixa *protein*. Per exemple, en un experiment podem haver cristalitzat només el 40% d'una proteina mentre que amb una altra tècnica se n'ha obtingut el 75%, o bé en el moment de la cristalització aquesta proteïna estava formant un enllaç amb una molècula i s'ha guardat la informació de tot l'enllaç...

4.4.1.4. Snapshot

En els *dockings*, les proteïnes segueixen una trajectòria des d'un punt d'origen fins a un punt final (quan formen l'enllaç). Durant aquesta trajectòria, les coordenades de la proteïna van variant. Així doncs, durant la trajectòria hi han diferents instàncies de la proteïna, cadascuna amb unes coordenades diferents. S'anomena *snapshot* a aquestes instàncies. Un conjunt de *snapshots* és un subgrup de les molècules fetes servides en una simulació, i que han estat seleccionades amb un criteri o estratègia determinat(strategy). Les coordenades dels *snapshots* vindran determinades per l'instant de temps en el que es va produir (*time*).

un set de snapshot es un subgrup de la simulació seleccionat amb un “criteri”/strategy determinat.

4.4.1.5. Chain

Les *chains* serveixen per identificar els diferents components que hi ha en una estructura. Venen representades per caràcters(*chain*) únics per cada estructura. Per exemple, una estructura podria estar conformada per una proteïna iun fàrmac. En aquest cas, podem dir que la proteïna és la cadena ‘A’ de l’estructura i el fàrmac és la cadena ‘B’.

4.4.1.6. SmallMolecule

S’anomena *small molecules* als compostos amb un pes molecular baix, i en el camp del descobriment de fàrmacs queda restringit als compostos que alteren l’activitat d’una proteïna quan s’uneix a aquesta. Així doncs, els fàrmacs estan formats per conjunts de *small molecules*.

Aquestes *small molecules* queden identificades per un nom (*name*) i tenen un conjunt de propietats físiques, que són utilitzades per discernir l’activitat que generen quan s’uneixen a les dianes. D’aquestes propietats, n’hi ha unes quantes que tenen més interès en el cas dels *dockings*, ja que s’ha observat que són més comuns en fàrmacs actius. Aquestes propietats són:

- Pes molecular (*molW*), és a dir, la massa de la molècula

- Nombre d’àtoms (*nAtoms*) que conformen la molècula

- Refractivitat molar (*refrac*), índex que determina la polarització

- Superfície polar (*polarSurf*), que és la suma de totes les superfícies de tots els àtoms polars (oxigen i nitrogen) a més dels hidrogens que tinguin enganxats.

- Coeficient de partició (*logP*), índex

Donat que existeixen una gran quantitat de descriptors (fins a milers), els menys utililtzats s’emmagatzemaran en disc (*folder*).

Per altra banda, s'utilitza la notació *SMILES* (Simplified Molecular-Input Line-Entry System [20]) per descriure l'estructura química de les molècules. Aquesta notació utilitza cadenes de caràcters per tal de descriure les molècules. Mitjançant editors de molècules, es pot utilitzar aquesta cadena per tal d'obtenir representacions en dues o tres dimensions de les molècules.

4.4.1.7. Activity

L'activitat farmacològica (o simplement activitat) és un terme utilitzat per determinar si un compost s'arriba a adherir a una proteïna. Aquesta activitat es pot obtenir a través de diferents fonts (*source*): realitzant experiments en un laboratori, mitjançant simulacions de dockings, a partir de la recerca en la literatura científica...

4.4.1.8. Docking

Tal com s'ha comentat anteriorment, s'anomena *docking* a l'enllaç que format per proteïnes i molècules. La fortalesa d'aquest enllaç és quantificada per l'energia necessària per trencar-lo (*value*).

4.4.1.9. Subset

Els *subsets* són subconjunts de *Small Molecules*. L'objectiu principal d'aquests és aconseguir agrupar molècules (seguint qualsevol criteri) i posteriorment utilitzar-les en algun experiment. Amb això s'aconsegueixen dues coses: per una banda, es redueix considerablement el nombre de molècules involucrades en els experiments, fent que el temps invertit disminueixi dràsticament; per altra banda, els criteris d'agrupament poden fer que els resultats obtinguts siguin millors (per exemple agrupant les molècules amb un resultat de *docking* elevat).

A més a més, els *subsets* també poden servir per a aprofitar resultats d'experiments en el futur. Això també fa que mitjançant l'aplicació el temps necessari per a realitzar aquests experiments sigui menor.

4.4.2. Base de dades relacional

4.4.2.1. Diagrama

En la següent figura es mostra l'esquema de la base de dades, dissenyat a partir de l'esquema conceptual per ser accedida des de l'aplicació. Els identificadors i taules intermitges serveixen per establir les relacions entre els conceptes.

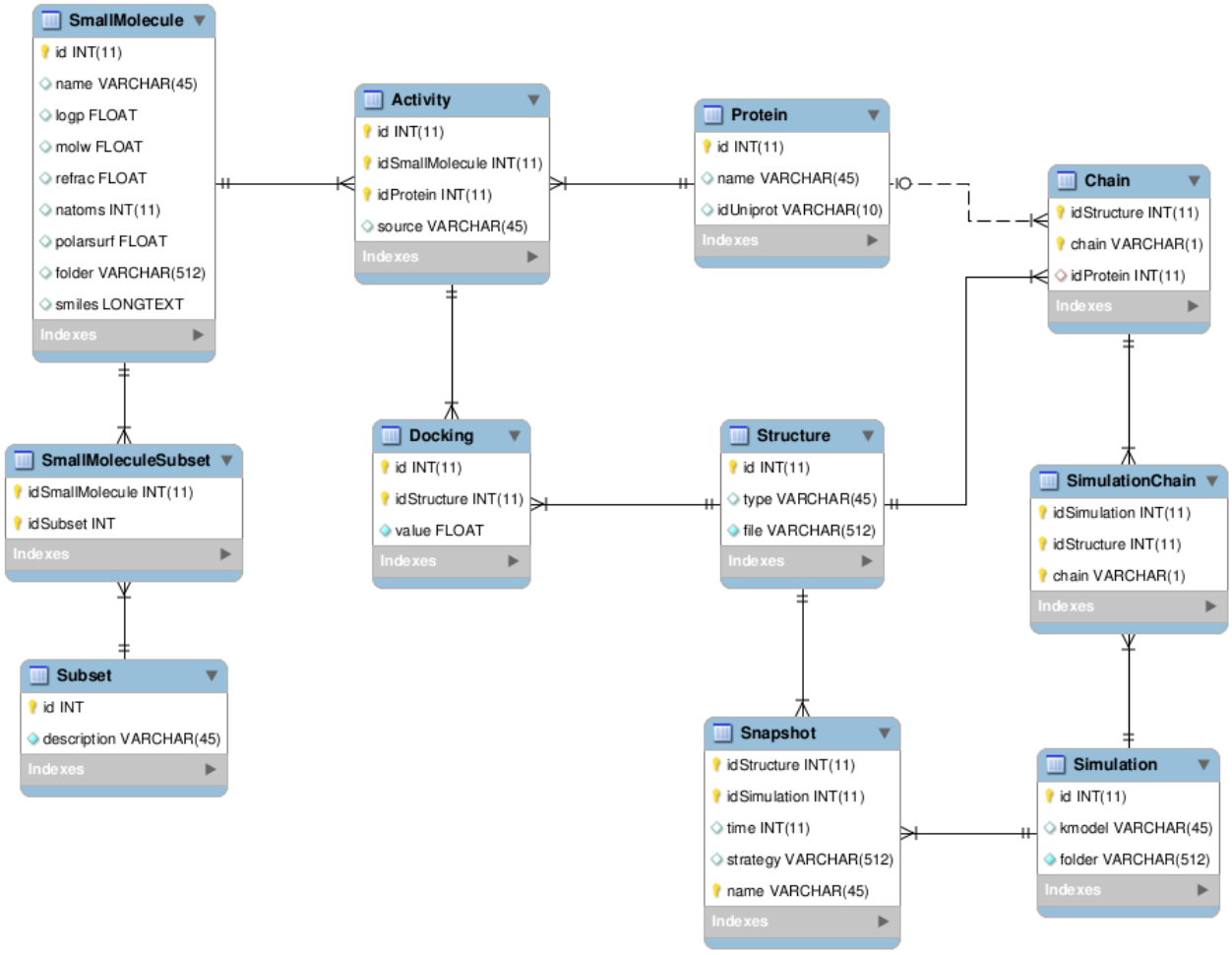


Figura: Esquema de la base de dades relacional

4.4.2.2. Inserció de dades

Per tal de comprovar el rendiment de la base de dades, s'ha inserit informació provinent de 5 simulacions de *docking*. A continuació es detallen les cardinalitats de les taules un cop inserides totes les dades:

Taula	**Nombre d'elements**
SmallMolecule	37.079.881
Protein	4
Structure	30
Snapshot	125
Simulation	5
Chain	42
SimulationChain	36
Activity	111.186
Docking	111.186
Subset	1
SmallMoleculeSubset	10.000

La taula SmallMolecule és la que té més elements donat que s'han volcat totes les molècules existents a Zinc[19], una base de dades de molècules de lliure accés i àmpliament utilitzada per la comunitat.

Per a comprovar que la base de dades dissenyada funcionava correctament i que aquesta oferia grans avantatges respecte el sistema de fitxers utilitzat fins llavors, s'han realitzat un seguit de consultes. Les consultes més interessants de cara als usuaris han estat:

- Obtenir totes les molècules que tenien activitat contra una diana
- Obtenir totes les molècules que tenien activitat contra més d'una diana
- Ordenar les molècules segons els resultats de *docking* obtinguts contra una diana

Fins al moment, aquest tipus de consultes no es podien realitzar d'una manera eficient, ja que totes les dades s'emmagatzemaven en fitxers en disc. La integració amb la base de dades soluciona aquest problema, a més d'oferir altres avantatges.

En una següent iteració del desenvolupament de l'aplicació caldria implementar la funcionalitat de treballar amb aquesta base de dades des de l'aplicació. Això pot donar pas al sorgiment de noves funcionalitats que millorin substancialment la utilitat de l'eina.

5. Organització

En aquest apartat es parlarà de com s'ha organitzat el projecte: es descriuran les etapes en les quals s'ha dividit, es donarà la planificació inicial i com aquesta s'ha modificat al final, es farà un estudi del temps invertit en cada etapa, es comentarà la metodologia que s'ha seguit i les eines que s'han utilitzat tant per desenvolupar el projecte com per organitzar-lo d'una manera més eficient.

5.1 Etapes del projecte

En l'eina SEABED s'ha utilitzat la metodologia de desenvolupament iterativa i creixent. En aquesta metodologia, el desenvolupament queda dividit en iteracions. Cadascuna de les iteracions consisteix en el desenvolupament de noves funcionalitats, sempre aprofitant el que s'ha desenvolupat en iteracions prèvies (d'aquí el nom de creixent).

Podem dividir el desenvolupament de l'aplicació en 4 etapes diferents: etapa prèvia, etapa d'anàlisi, etapa de desenvolupament i etapa de testeig. En el diagrama següent es mostren cadascuna de les etapes i les transicions que hi han entre elles.

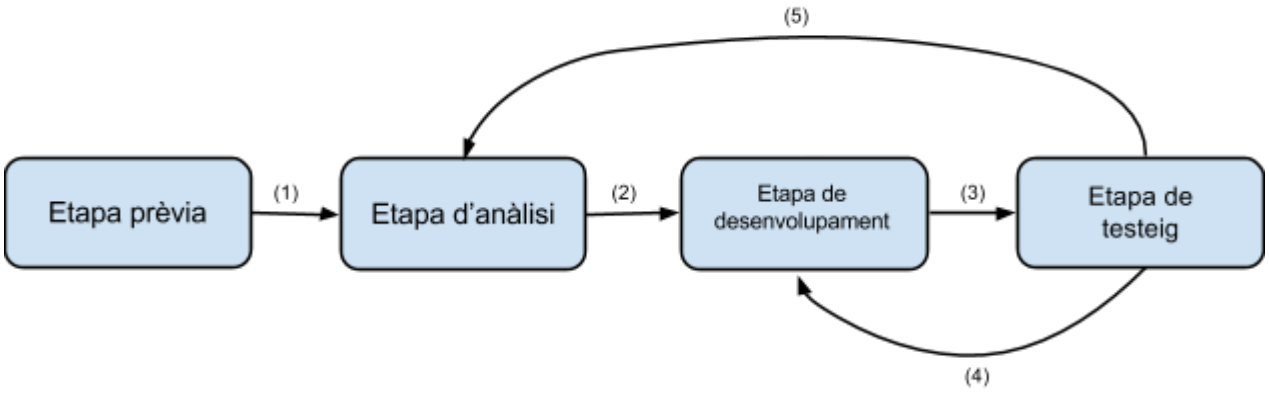


Figura: Etapes de la metodologia de desenvolupament creixent

Les iteracions queden definides des de que es comença l'etapa d'anàlisi fins que es detecta una nova funcionalitat per a l'aplicació.

5.1.1. Etapa prèvia

S'anomena etapa prèvia al temps invertit en l'estudi dels diferents components que conformaran l'aplicació. Les principals tasques desenvolupades en aquesta etapa han estat:

- Estudi de les llibreries biomèdiques a utilitzar en l'aplicació
- Estudi del framework *Vaadin*
- Creació de l'entorn de treball a utilitzar durant el projecte

5.1.2. Etapa d'anàlisi

Durant l'etapa d'anàlisi s'estudien les noves funcionalitats a implementar en l'aplicació. Aquestes noves funcionalitats poden venir donades per dues vies: pels requeriments inicials de l'aplicació o bé per noves necessitats detectades durant l'etapa de testeig.

En el primer cas (transició 1 a la figura anterior), després de l'etapa prèvia es va procedir a l'anàlisi dels requeriments (funcionals i no funcionals) de l'aplicació. En aquest moment es va implementar l'esquelet de l'aplicació, i es va realitzar un primer esbós de com es volia que fós. Aquests requeriments inicials eren els que es consideraven fonamentals per a que l'aplicació tingués certa utilitat. Alguns d'aquests eren: poder realitzar els *workflows*, mantenir un sistema d'usuaris i projectes, permetre treballar en diferents *workflows* al mateix temps, etc. Com es pot veure, són requeriments sense els quals l'aplicació no generaria cap interès als seus futurs usuaris.

Els requeriments sorgits de l'etapa de testeig (transició 5 en la figura anterior) estan orientats a millorar la usabilitat general de l'aplicació, o a corregir *bugs* greus. Alguns d'aquests requeriments han estat: poder crear un model utilitzant uns resultats d'un *docking* finalitzat, mostrar més informació dels models creats (paràmetres i *ROC Curve*),

poder crear els models amb tots els algorismes a la vegada (per evitar tornar a calcular els descriptors)... Com es pot veure, aquestes funcionalitats no són imprescindibles per al funcionament de l'aplicació, però ofereixen una gran millora en la usabilitat d'aquesta, ja que s'aconsegueix que els usuaris no realitzin el treball per duplicat. Aquests requeriments apareixen quan l'eina és utilitzada en casos reals, ja que en l'etapa inicial no es poden preveure totes les necessitats futures de l'aplicació.

5.1.3. Etapa de desenvolupament

Al finalitzar l'etapa d'anàlisi (transició 2 a la figura anterior), es disposa d'un conjunt de casos d'ús que cal implementar. Aquesta tasca es durà a terme en l'etapa de desenvolupament.

Donat que aquesta és una de les etapes que més temps consumeix, juntament amb la de testeig, s'intentarà implementar els casos d'ús aprofitant funcionalitats ja existents, ja sigui en aquesta aplicació o en altres aplicacions que fan servir el *framework Vaadin.* D'aquesta manera es redueix el temps destinat a aquesta etapa, ja que s'entén que les parts que es reaprofiten funcionen correctament.

5.1.4. Etapa de testeig

En aquesta etapa és on es comprova que les noves funcionalitats que s'han implementat a l'etapa anterior funcionen correctament. Encara que al llarg de l'etapa anterior s'hagin realitzat un petit conjunt de proves per garantir que el que s'estava implementant funcionava, és en aquesta etapa en la que realment es posa a prova l'aplicació amb dades reals.

Així doncs, amb l'ajuda d'altres membres del departament, en aquesta etapa es realitzen un seguit de proves amb dades contrastades per tal d'assegurar que tot el conjunt de l'aplicació funciona correctament i que s'obtenen els resultats esperats.

Quan finalitza aquesta etapa es poden donar tres escenaris:

a) Totes les proves resulten satisfactòries, i es pot passar a desenvolupar noves funcionalitats a l'etapa de desenvolupament

b) S’observa un comportament inesperat en alguna de les noves funcionalitats, i per tant es torna a l’etapa de desenvolupament per corregir-la (transició 4 de la figura anterior).

c) Es descobreix una possible millora a l’aplicació arran de l’ús de l’eina, i es torna a l’etapa d’anàlisi per tal d’estudiar amb més detall aquesta millora i si val la pena incloure-la a l’eina (transició 5 a la figura anterior)

5.2. Planificació del projecte

En el diagrama de Gantt següent es mostra la planificació inicial del projecte.

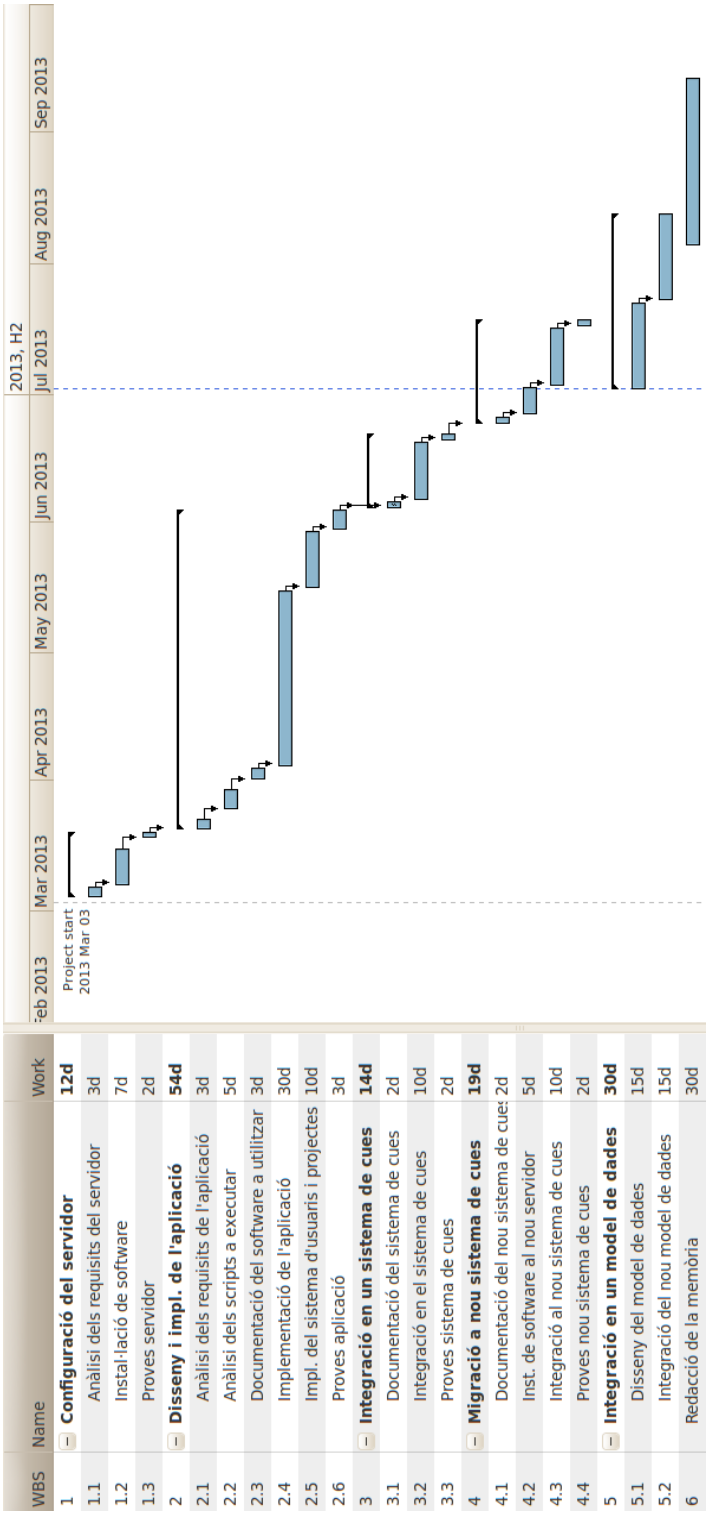


Figura: Planificació inicial del projecte

No obstant, aquesta planificació inicial s'ha vist alterada degut a diversos factors. Primerament, al llarg del desenvolupament del projecte s'han implementat funcionalitats que no estaven previstes originalment. Algunes d'aquestes noves funcionalitats són:

- Substitució de les llibreries que s'utilitzaven originalment per a realitzar els càlculs per unes amb més funcionalitats. Aquestes llibreries tenien unes dependències diferents a les originals, i per tant s'ha hagut d'instal·lar nou programari en els servidors per tal de poder-les utilitzar.
- Modificació dels *workflows* de l'aplicació: quan s'ha donat ús a l'eina han sorgit nous requeriments per tal de millorar la seva usabilitat. La majoria d'aquestes millores feien referència als fluxos de treball (*workflows*). Es va detectar que alguns passos eren repetits múltiples vegades en un mateix *workflow*, i es van realitzar modificacions tant a la interfície com a la lògica de l'aplicació per tal de minimitzar aquest fet.
- Inserció a la base de dades de resultats provinents d'experiments previs.

En el diagrama de *Gantt* següent es mostra el temps invertit finalment en cada etapa del projecte.

WBS	Name	Work
1	**Configuració del servidor**	**12d**
1.1	Anàlisi dels requisits del servidor	3d
1.2	Instal·lació de software	7d
1.3	Proves servidor	2d
2	**Disseny i impl. de l'aplicació**	**54d**
2.1	Anàlisi dels requisits de l'aplicació	3d
2.2	Anàlisi dels scripts a executar	5d
2.3	Documentació del software a utilitzar	3d
2.4	Implementació de l'aplicació	30d
2.5	Impl. del sistema d'usuaris i projectes	10d
2.6	Proves aplicació	3d
3	**Integració en un sistema de cues**	**14d**
3.1	Documentació del sistema de cues	2d
3.2	Integració en el sistema de cues	10d
3.3	Proves sistema de cues	2d
4	**Migració a nou sistema de cues**	**19d**
4.1	Documentació del nou sistema de cues	2d
4.2	Inst. de software al nou servidor	5d
4.3	Integració al nou sistema de cues	10d
4.4	Proves nou sistema de cues	2d
5	**Integració en una base de dades**	**29d**
5.1	Disseny del model de dades	15d
5.2	Creació de la base de dades relacional	4d
5.3	Inserció de dades a la bd relacional	10d
6	Redacció de la memòria	40d

Figura: Planificació final del projecte

5.3. Eines utilitzades

En aquest apartat es donarà una breu descripció de les eines utilitzades al llarg del projecte per dur a terme les múltiples tasques que l'han conformat.

5.3.1. Desenvolupament

5.3.1.1. Eclipse

S'ha utilitzat Eclipse [22] com a entorn de programació principal. Eclipse és un *Integrated Development Environment(IDE)* molt popular degut a la gran quantitat de *plug-ins* que ofereix i la seva facilitat d'ús. El principal avantatge que tenen els *IDE* respecte als editors de text és que s'integren en una sola eina molts components necessaris per al desenvolupament de *software*: compiladors, *debuggers*, intèrprets, etc. No obstant, això implica que els *IDE*s consumeixen molts més recursos que un editor de text senzill, fent que aquests no puguin ser utilitzats en màquines poc potents.

En el cas de la nostra aplicació, un gran avantatge que oferia *Eclipse* és que els desenvolupadors del *framework Vaadin* van crear un *plugin* per tal de poder gestionar els projectes des de la pròpia aplicació. D'aquesta manera, un cop instal·lat el *plugin,* ha estat molt senzill crear i gestionar la configuració de l'aplicació.

5.3.1.2. Sublime Text Editor

He utilitzat l'editor de text *Sublime Text Editor* [23] per desenvolupar petits *scripts* d'inserció a la base de dades. He optat per utilitzar aquest editor donat que no es requeria d'un entorn tan complex com en el cas anterior, i a més per què ofereix algunes de les funcionalitats més pròpies dels *IDEs* sense consumir tants recursos.

5.3.2. Gestió de la base de dades

5.3.2.1. MySQL Workbench

MySQL Workbench [24] és una eina que permet crear models de dades, ofereix un entorn de programació *SQL*, i a més disposa d'eines d'administració de les bases de dades: servidors, usuaris, *backups*...I tot això des de una interfície gràfica.

S'ha escollit aquesta eina donat que agilitza molt el procés de crear una base de dades, ja que es pot crear un model de dades visualment i posteriorment obtenir un *script*

SQL que creï la base de dades. Aquest mètode és molt més senzill i intuïtiu que no pas si es creés aquest *script* manualment.

5.3.3. Control de versions

5.3.3.1. Git

Git[25] és un dels sistemes de control de versions més utilitzats actualment, juntament amb altres com Subversion o CVS. L'objectiu dels sistemes de control de versions és el d'organitzar les diferents versions del codi que es desenvolupen en qualsevol aplicació. Gràcies a aquests sistemes, es poden revertir canvis en el codi, es pot realitzar un seguiment del treball realitzat pels diferents membres dintre d'un mateix equip, etc.

Com s'ha comentat en un apartat anterior, el servidor principal disposa d'un repositori on s'hi van guardant tots els projectes desenvolupats al grup. Així doncs, s'ha afegit el codi de l'aplicació *SEABED* aquest repositori per facilitar-ne el control del codi.

A més a més, la utilització de *Git* ha permès que diferents membres del grup desenvolupin noves funcionalitats en paral·lel, reduïnt el temps invertit en l'etapa de desenvolupament.

5.4. Metodologia de desenvolupament

En aquest apartat es parlarà de la metodologia que s'ha seguit a l'etapa de desenvolupament del projecte. Es descriuran els entorns on s'ha desenvolupat l'aplicació i es detallaran els passos seguits des de que es comença a desenvolupar una nova funcionalitat fins que els usuaris la poden utilitzar des de l'aplicació.

5.4.1. Entorns de l'aplicació

Anomenem entorns de l'aplicació a les màquines assignades a les diferents etapes del procés de desenvolupament de l'aplicació *SEABED*. Distingim dos entorns: desenvolupament i producció.

5.4.1.1. Entorn de desenvolupament

Aquest és l'entorn que tenen els desenvolupadors en local a les seves màquines. Així doncs, cada desenvolupador tindrà una versió de l'aplicació, sobre la que es desenvoluparan les noves funcionalitats d'aquesta.

Les versions en local no són visibles pels usuaris de l'aplicació ja que sofreixen canvis constants durant l'etapa de desenvolupament, i per tant no són estables i sovint es van trobant *bugs* que es solucionen en el mateix entorn.

5.4.1.2. Entorn de producció

Anomenem entorn de producció al que es troba en el servidor principal. En aquest entorn s'hi troba la versió estable de l'aplicació, la qual és accedida pels usuaris. Així doncs, en aquesta es dóna prioritat a l'estabilitat de l'aplicació en detriment del temps transcorregut entre l'aparició de noves funcionalitats.

Si es detectés algun error en aquesta versió, aquest es corregiria primer en l'entorn de desenvolupament i, un cop corregit l'error, es pujaria la nova versió a aquest entorn. D'aquesta manera, la versió de l'apliació en aquest entorn mai és modificada directament pels desenvolupadors, si no que es substituïda per una versió provinent de l'entorn de desenvolupament.

Aquesta metodologia permet que els usuaris es trobin amb versions més robustes de l'aplicació, que només es veuen modificades quan s'hi afegeixen noves funcionalitats un cop aquestes han passat satisfactòriament per una etapa de proves en l'entorn de desenvolupament.

Normalment, en el desenvolupament de *software* hi ha (com a mínim) un altre entorn entre els dos entorns dels que em parlat, anomenat entorn de test. En aquest, l'objectiu és detectar errors en les noves funcionalitats implementades en l'entorn de desenvolupament abans de ser pujades a l'entorn de producció, per tal de reduir el nombre d'errors trobats en l'aplicació accessible pels usuaris.

Idealment, les màquines de l'entorn de test són el més semblants a les màquines de l'entorn de producció, tant en *hardware* com en *software*. Això és així per evitar un mal funcionament de l'aplicació degut a que les màquines de l'entorn de desenvolupament i producció utilitzin versions diferents de programari, o bé tinguin una arquitectura molt diferent.

No obstant, en el cas de l'aplicació *SEABED*, les màquines de desenvolupament i les de producció utilitzen el mateix *software* (tant el sistema operatiu com les aplicacions), així doncs, el fet de tenir un entorn exclusiu de test no és tan indispensable. Encara així, es pot considerar com una millora en el futur el fet d'establir un entorn de test per a l'aplicació, de manera que el funcionament de les noves versions de l'aplicació fos comprovat per diferents membres del grup abans de ser pujat a l'entorn de producció.

5.4.2. Metodologia

Un cop s'han vist els dos entorns dels que es disposa a l'aplicació, s'analitzaran els passos que es segueixen des de que s'obté una nova funcionalitat a desenvolupar des de l'etapa d'anàlisi fins que aquesta es posa a disponibilitat dels usuaris.

Així doncs, els passos a seguir són:

1. A l'etapa d'anàlisi (ja sigui per un requeriment inicial o per una necessitat detectada amb l'ús de l'eina) es decideix que una funcionalitat s'ha d'implementar per donar més valor a l'aplicació.

2. El desenvolupador prendrà com a referència la versió de l'aplicació que es troba a l'entorn de producció, i començarà a implementar la nova funcionalitat.

3. A mesura que el desenvolupador va implementant la nova funcionalitat, aquest comprova que l'aplicació continua funcionant amb dades de mostra.

4. Quan s'ha finalitzat el desenvolupament de la nova funcionalitat i els resultats de les proves és satisfactori, l'usuari puja els canvis al repositori del *Git* on es troba la versió estable de l'aplicació. Utilitzant la terminologia del *Git*, es fa un *push* del codi des de la màquina local del desenvolupador fins al repositori remot.

5. Un cop s'ha actualitzat el repositori remot amb la nova funcionalitat, un *script* copiarà automàticament els fitxers que s'han modificat a la carpeta on es troba el codi executat pel *Tomcat*, i que per tant és el codi de l'aplicació que utilitzen els usuaris.

6. Finalment, cal reiniciar el *Tomcat* en el servidor principal per tal que els canvis facin efecte. Es pot pensar que els usuaris poden perdre treball que estiguin realitzant en aquest moment, però això no és així: tots els càlculs dels *workflows* s'executen en màquines alienes a l'estat del *Tomcat* i, donat que el *QueueDaemon* continuarà corrent, s'avisarà els usuaris quan finalitzin els seus processos.

Aquests són els passos que cal seguir quan la implementació de noves funcionalitats només requereixi modificacions en l'aplicació. No obstant, si cal realitzar canvis en el *Queue System*, cal modificar els passos explicats lleugerament: els primers quatre passos continuen igual, i els dos últims passen a ser:

5. Un cop s'ha actualitzat el repositori remot amb la nova funcionalitat, un *script* copiarà automàticament els fitxers que s'han modificat a la carpeta on es troba el codi del *Queue Daemon* que s'executa a l'aplicació.

6. Finalment, cal reiniciar el *Queue Daemon* en el servidor principal per tal que els canvis facin efecte. Donat que l'estat dels treballs és emmagatzemat en una base de dades, quan es reinicïi el *Queue Daemon* es podrà continuar comprovant l'estat dels treballs.

En el següent diagrama es mostren els passos seguits pels desenvolupadors per tal d'implementar una nova funcionalitat i fer-la accessible als usuaris.

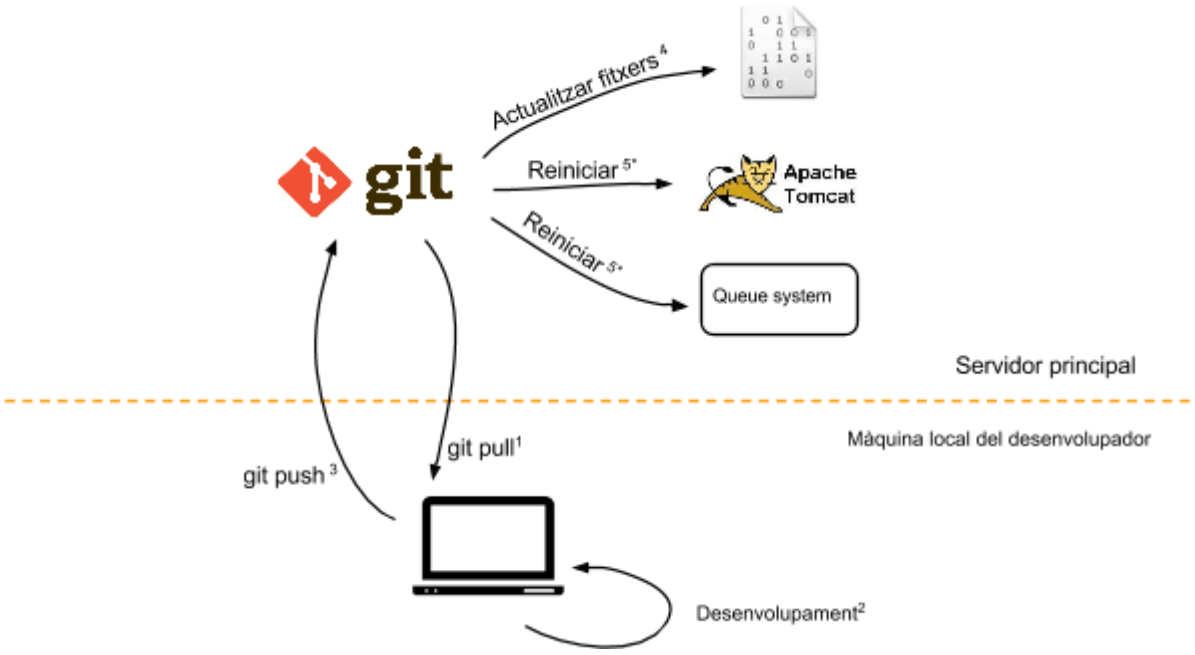


Figura: Passos seguits en el desenvolupament de l'aplicació

Les fletxes que s'originen al repositori git i van cap a la dreta fan referència a que, després del pas 3, hi ha un *hook* que realitza els passos 4 i 5. En la terminologia del *Git*,

un *hook* permet executar *scripts* després de produir-se un event determinat. En el nostre cas, s'executa un *script* que actualitza els fitxers modificats i reinicia el *tomcat* o el *Queue System* quan els desenvolupadors pugen canvis al repositori (git pull).

Una possible millora per al futur consistiria en comprovar l'estat del *Tomcat* i del *Queue Daemon* abans de reiniciar-los, per tal d'esperar a que no s'estigui transferint cap fitxer a l'aplicació.

5.4.3. Iteracions

En aquest apartat es detallaran les tasques realitzades en cadascuna de les iteracions en les que s'ha dividit el projecte.

5.4.3.1. Iteració 1

En la primera iteració de l'aplicació es van tractar els punts més assequibles d'aquesta:

- Creació de l'aplicació i *deploy* a un servidor web
- Disseny de l'estructura de fitxers de l'aplicació
- Disseny de la interfície de l'aplicació
- Implementació de funcionalitats bàsiques

Primerament, es va crear l'aplicació inicial de *Vaadin*. Posteriorment, es va desplegar aquesta aplicació sobre el servidor *Tomcat* per tal de poder ser accedida des de qualsevol ordinador amb accés a internet.

Un cop es va disposar d'una aplicació accessible via web, es va procedir al disseny de totes les finestres de l'aplicació. En aquest punt encara no s'executaven els càlculs que es necessiten per a realitzar els *workflows*, si no que només s'havien implementat els enllaços entre totes les finestres. A més a més, també es va implementar la funcionalitat de pujar fitxers a l'aplicació i l'edició dels fitxers pujats.

5.4.3.2. Iteració 2: Integració de les llibreries de *scripts* inicials

Quan ja es disposava de l'esquelet de l'aplicació, es va procedir a realitzar les següents tasques:

- Creació de *wrappers* de les comandes a executar en l'aplicació
- Connexió de l'aplicació dels *wrappers* amb l'aplicació per tal de poder connectar-la amb les llibreries de *scripts*

Donat que les llibreries de *scripts* només eren accessibles a través de la terminal, es van tenir que implementar *wrappers* per tal de poder ser cridades des de l'aplicació. Aquests *wrappers* són classes en Java que disposen de la sintaxi i els paràmetres per fer servir qualsevol de les llibreries.

Un cop es van implementar aquestes crides, es va procedir a la modificació de l'aplicació per tal d'executar-les. De moment aquestes crides s'executaven en la mateixa màquina on estava corrent l'aplicació, bloquejant els usuaris durant la seva execució.

5.4.3.3. Iteració 3: Integració de l'aplicació en un sistema de cues

Les tasques desenvolupades en aquesta iteració han estat:

- Creació de les classes necessàries per interaccionar amb el *Queue System*.
- Modificació de la interfície original per tal de mostrar als usuaris informació sobre l'estat de les seves execucions.
- Implementació del sistema de notificacions als usuaris a través del correu.

Un cop ja es podien cridar les llibreries des de l'aplicació, es va procedir a integrar aquestes crides en *Queue System*, un sistema de cues ja existent utilitzat en altres aplicacions del grup.

Es van aprofitar els *wrappers* desenvolupats anteriorment i es van implementar els *Abstract jobs* necessaris per a executar les crides en el *Queue System*. Com en aquest

punt totes les crides es llençaven a la cua *SGE*, no va ser necessari implementar cap altra funcionalitat referent al gestor de les cues.

El següent pas va ser modificar la interfície original per permetre als usuaris treballar en més d'un *workflow* a la vegada i reprendre un *workflow* en un pas intermedi quan un càlcul havia finalitzat després de que l'usuari hagués sortit de l'aplicació. Aquesta funcionalitat va fer que la valoració de l'eina s'incrementés considerablement, ja que fins a les hores els usuaris no podien sortir de l'aplicació si no volien perdre el seu treball. Tenint en compte que alguns dels càlculs poden tardar dies a executar-se, es fa evident que abans d'aquesta iteració l'eina no era usable.

Finalment es va procedir a implementar les notificacions per correu electrònic als usuaris. Amb això s'aconsegueix que els usuaris no hagin d'estar pendents de si ha finalitzat algun procés llarg o no, si no que reben una notificació quan poden reprendre un *workflow* qualsevol.

5.4.3.4. Iteració 4: Modificació de les llibreries utilitzades i integració en el sistema de cues de Life Sciences

En aquest moment ja es disposava d'una aplicació d'utilitat per als usuaris. No obstant, van aparèixer noves funcionalitats no previstes al començament del projecte. Les noves tasques a realitzar van ser:

- Modificació de les llibreries utilitzades des de l'aplicació
- Integració de l'aplicació amb el sistema de cues del cluster de Life Sciences

Per una banda, paral·lelament al desenvolupament de l'aplicació SEABED, altres membres del grup van anar desenvolupant noves llibreries més completes que les ja existents. Aquestes llibreries estaven desenvolupades amb el llenguatge de programació R, a diferència de les llibreries originals que utilitzaven Python. Un cop es va comprovar que amb les noves llibreries s'obtenien millors resultats que amb les originals es va decidir utilitzar-les també a l'aplicació. Això va fer necessari crear nous *wrappers* per a les crides i nous *jobs* per tal de ser processades pel Queue System.

Per altra banda, en aquest punt es va tenir accés al Cluster de Life Science. Donat que aquest disposa d'una capacitat de càlcul molt més elevada que les *work stations* sobre

les que es realitzaven els càlculs fins a les hores, es va decidir que els processos més costosos (*dockings*) utilitzessin aquest cluster per tal de reduir el temps invertit en els experiments. Això va fer necessari instal·lar les llibreries i software necessàries per a llençar els dockings a les noves màquines. Aquest fet no va ser necessari quan es van integrar les crides originals, ja que el servidor on es troba l'aplicació ja disposa d'aquest programari.

Donat que el sistema de cues emprat al Cluster no era el mateix que l'original, va ser necessari implementar noves funcionalitats al *Queue System* per tal de poder utilitzar tots dos sistemes de cues a la vegada. Concretament, va ser neecssari implementar una classe abstracta per a referenciar aquesta nova cua. Finalment, també va ser necessari implementar el protocol per a transferir els fitxers entre els dos fitxers, tant abans com després de l'execució dels processos.

5.4.3.5. Iteració 5: Base de dades

En aquesta iteració es va decidir dissenyar una base de dades per poder analitzar tota la informació que es tractava des de l'aplicació. Concretament, les tasques desenvolupades van ser:

- Disseny del model de dades de l'aplicació
- Disseny d'una base de dades relacional
- Volcat d'experiments a la base de dades
- Implementació de *wrappers* per a poder accedir a la base de dades des de l'aplicació

Quan ja s'havia comprovat la millora que havia suposat el trasllat de part dels càlculs de l'aplicació al Cluster, va sorgir la necessitat d'integrar una base de dades a l'aplicació. Amb aquesta funcionalitat, s'aconseguirien dues coses: no repetir els càlculs i realitzar consultes sobre les dades.

Amb aquest objectiu, es va realitzar un disseny del model de dades de l'aplicació amb l'ajuda d'un altre grup de recerca del BSC. Aquest grup pertany al departament de Computer Science del centre, i està enfocat a la recerca en sistemes d'emmagatzemament. Fruit d'aquesta col·laboració va sorgir el diagrama del model de dades.

A partir d'aquest diagrama, es va procedir a fer un disseny d'una base de dades relacional

que posteriorment seria accedida des de l'aplicació. Un cop es van crear les taules de la base de dades, es va decidir que es volcaria informació provinent d'experiments previs per tal de comprovar la correctesa d'aquesta.

Donat que els resultats d'aquests experiments es trobaven emmagatzemats en una gran quantitat de fitxers, va ser necessari implementar *scripts* per tal d'extreure la informació útil de tots aquests fitxers per ser processada per la base de dades posteriorment.

Quan es disposava de tota la informació necessària, es va procedir a realitzar consultes sobre la base de dades per tal de calcular el rendiment d'aquesta. En aquest punt va quedar clar que gràcies a aquesta era possible extreure molta informació nova dels experiments, molt difícil d'aconseguir fins llavors des dels fitxers.

Finalment, es van implementar les classes Java que servirien per posteriorment poder llençar aquestes consultes des de la mateixa aplicació. No obstant, encara que es pugui accedir a les dades, encara falta per implementar aquestes consultes i la part de la interfície de l'aplicació que permeti realitzar-les. Aquesta hauria de ser la propera funcionalitat a desenvolupar en el projecte.

6.Conclusions

6.1. Futures millores

Al llarg del desenvolupament, i sobretot en l'etapa en la qual usuaris interns han anat donant ús real a l'eina, s'han trobat diferents aspectes que caldria millorar en una iteració futura de l'aplicació. En aquest apartat s'han reunit algunes de les propostes més interessants de cara a ser implementades en un futur.

6.1.1. Integració amb la base de dades

Tal com s'ha vist en apartats anteriors, en el transcurs del desenvolupament del projecte s'ha realitzat el disseny d'un model conceptual de dades per a l'aplicació. A més a més, s'ha traduït aquest model a una base de dades relacional, fent servir MySQL com a gestor de la base de dades per tal d'aprofitar components d'altres bases de dades existents dintre del grup.

Per altra banda, en aquesta base de dades s'han introduït resultats d'experiments realitzats amb anterioritat, per comprovar que el disseny era el correcte i que l'aplicació podria beneficiar-se de les dades emmagatzemades. A més a més s'han desenvolupat les classes de Java (*wrappers*), que permeten accedir a les diferents taules de la base de dades des de la pròpia aplicació.

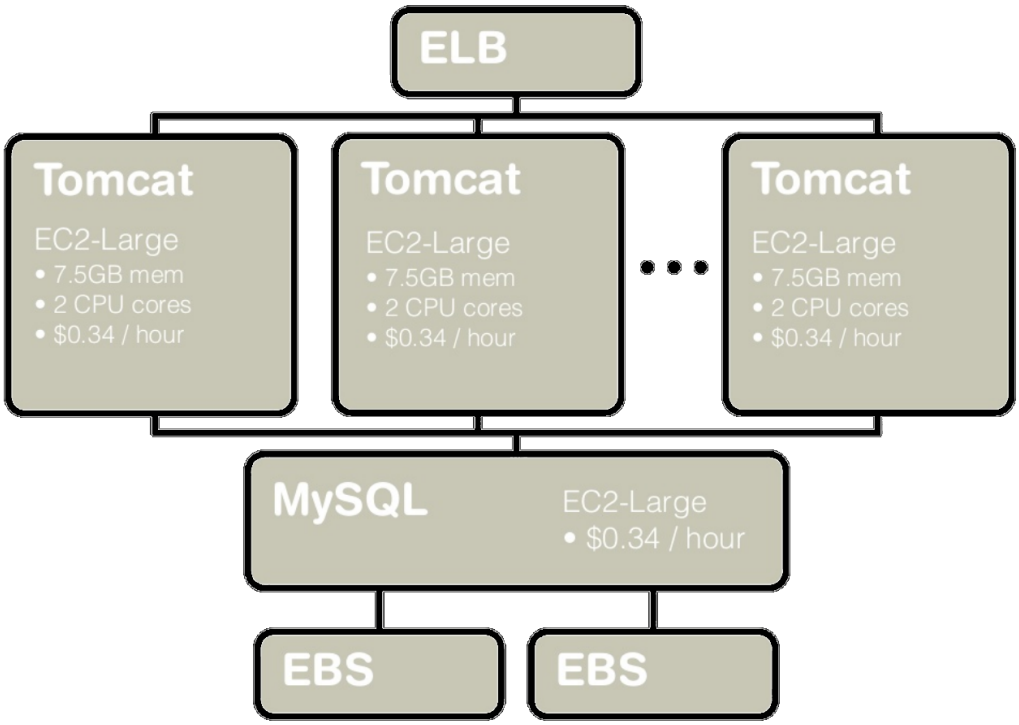
No obstant, aquesta funcionalitat encara no s'ha explotat donat que, per falta de temps, no s'han desenvolupat les consultes específiques per tal de treballar amb la base de dades.

Així doncs, una de les millores més prioritàries hauria de ser estudiar les consultes més útils per als usuaris i modificar la interfície per a que aquestes puguin ser cridades des de l'aplicació. Amb això, els usuaris de SEABED es beneficiarien de disposar d'informació que els hi evitaria repetir processos (per exemple emmagatzemant els descriptors calculats), a més de disposar d'informació extreta de bases de dades externes o d'experiments ja realitzats (per exemple afegint col·leccions de molècules de bases de dades externes).

6.1.2. Implementació de millores en l'escalabilitat

Actualment, l'aplicació està corrent dins una sola màquina, fet que implica que, si en algun moment aquesta falla, no serà accessible. Per tal de millorar l'escalabilitat i la tolerància a fallades de l'aplicació, seria necessari realitzar proves de rendiment quan l'aplicació sigui utilitzada diàriament.

Intentant buscar una resposta a l'escalabilitat del framework Vaadin, els seus enginyers van realitzar uns experiments per tal de descobrir quins eren els límits de la seva eina [26]. Cal tenir en compte que les seves proves es van realitzar en un entorn bastant diferent al de la nostra aplicació, per tant els resultats obtinguts podrien no concordar amb el nostre cas en concret, encara que ens serviran per obtenir una idea en general. En el seu cas, la seva aplicació corria sobre instàncies d'Amazon EC2[27], amb un balancejador de càrrega per tal de repartir el volum de treball entre les diferents instàncies. A més a més, disposaven d'utilitats d'Amazon per treballar amb aquest tipus d'instàncies, com puguin ser eines de sincronització entre les diferents instàncies (Elastic Load Balancing) i entre les diferents bases de dades (Elastic Block Store). En la següent figura es mostra l'arquitectura de l'aplicació utilitzada en l'estudi.



Per tant, un dels primers passos a seguir per escalar la nostra aplicació consistiria en migrar-la a una arquitectura semblant a la proposada en l'exemple. Cal tenir en compte que els resultats de l'estudi conclouen que l'escalabilitat depèn en gran mesura del tipus d'aplicacions, i per tant caldrà estudiar quins punts de la nostra aplicació són els que tenen un impacte més elevat en l'escalabilitat del sistema. A priori, el fet de tenir un únic punt d'entrada a l'aplicació sembla ser el que més la perjudica en aquest aspecte.

Per altra banda, caldrà estudiar el codi de les llibreries que s'executen des de l'aplicació, per tal de paral·lelitzar-lo i d'aquesta manera reduir el temps d'execució dels càlculs i per tant poder donar suport a un nombre major d'usuaris.

Amb aquest objectiu en ment, seria interessant estudiar nous models de programació que ens aportin aquesta paral·lelització del codi. Un exemple d'aquests models és l'anomenat COMP Superscalar(COMPSs) [28], desenvolupat al BSC.

Les principals virtuds d'aquest model de programació són:

- Programació seqüencial, reduïnt els problemes que sorgeixen amb la programació en paral·lel: creació i sincronització entre processos, enviament de missatges entre processos, recuperació d'errors, etc.
- Independència de l'infraestructura, fet que afavoreix la portabilitat de les aplicacions
- Desenvolupat amb llibreries i components propis de Java, gràcies a això l'aprenentatge i implementació del model no resulta tan costós.

A més a més, recentment s'ha publicat una nova versió de COMPSs, integrant, entre d'altres, interoperabilitat amb múltiples sistemes cloud, com Emotive, Amazon EC2, OCCI[29], OpenNebula[30] i millores en l'escalabilitat i elasticitat amb aquestes plataformes. Així doncs, si l'actual arquitectura no pogués donar suport a tants usuaris al mateix temps es podria migrar cap a una d'aquestes plataformes per augmentar tant l'escalabilitat com la tolerància a fallades.

6.1.3. Implementació de noves funcionalitats i millora de la usabilitat

Donat que actualment l'aplicació SEABED és utilitzada internament, els casos d'ús han vingut definits pels membres del grup on s'ha desenvolupat. No obstant, és d'esperar que, un cop l'aplicació tingui un cert nombre d'usuaris, apareguin noves necessitats per als clients que no han estat detectades pels usuaris inicials, o que es trobin problemes de rendiment degut a l'accés simultani d'un nombre d'usuaris elevat.

Així doncs, amb l'objectiu de descobrir punts dèbils de l'eina o noves funcionalitats per millorar-la, es podria crear un compte de correu de contacte a la pàgina, on els usuaris poguessin donar la seva opinió sobre l'eina i suggerir possibles millores a tenir en compte.

El fet d'anar descobrint noves funcionalitats a mesura que es va fent servir l'eina s'ha fet palpable gràcies a la utilització d'aquesta en casos reals. Alguns exemples d'aquests casos d'ús poden ser:

- Poder compartir dades entre diferents projectes d'un mateix usuari
- Oferir la possibilitat als usuaris de seleccionar una etiqueta dintre dels seus SDFs que determini l'activitat de les molècules
- Quan s'està creant un model amb la metodologia PSAR, no considerar com a inactives aquelles molècules que estan molt a prop del rang establert per l'usuari

6.2. Conclusió personal

Un cop finalitzat el projecte, cal fer una retrospectiva per tal de valorar quins resultats s'han obtingut des dels seus orígens.

L'objectiu principal del projecte era el de facilitar l'accés a membres de la comunitat científica a unes llibreries utilitzades com a suport en el descobriment de nous fàrmacs potencials. Per tal de satisfer aquest objectiu, l'aplicació havia de ser lliurement accessible i usable, ja que d'aquesta manera no es requeriria que els seus usuaris tinguessin coneixements tècnics molt elevats.

Aquest objectiu s'ha vist complert i, a més a més, s'han desenvolupat algunes funcionalitats que han fet que l'eina sigui més útil que en el moment de la seva concepció. Això fa que la valoració personal d'aquesta part del projecte sigui molt positiva.

Per altra banda, el següent gran objectiu del projecte era el d'utilitzar un model de dades que permetés gestionar les que es feien servir des de l'aplicació i, a més a més, poder tractar aquestes dades per tal d'obtenir informació més rellevant dels experiments duts a terme. Per tal de fer-ho possible, s'ha realitzat el disseny d'una base de dades relacional a partir del model de dades de l'aplicació, i, posteriorment, s'han introduït els resultats d'uns experiments realitzats prèviament en el grup per tal de poder qualificar la utilitat d'aquesta base de dades. Els resultats d'aquest estudi han estat satisfactoris, ja que un cop importades les dades i mitjançant algunes consultes preparades, s'han pogut descobrir nous resultats que abans eren de difícil accés, ja que es trobaven emmagatzemats en fitxers i per tant no oferien la potència de les consultes en una base de dades relacional.

No obstant, l'objectiu d'integrar la base de dades amb l'aplicació no s'ha pogut dur a terme degut a la falta de temps, i això fa que la valoració en aquest aspecte no sigui tant positiva com en el cas anterior.

Finalment, crec que l'experiència de realitzar aquest projecte també m'ha resultat molt gratificant, ja que he tingut l'oportunitat d'aprendre moltes coses noves i treballar en un equip tan punter com el del BSC. Cal dir que el fet de no tenir gairebé coneixements en el camp biomèdic m'ha dificultat el desenvolupament del projecte, sobretot en les etapes

inicials. Si hagués disposat d'aquests coneixements, el temps invertit en estudiar les nocions bàsiques s'hagués pogut aprofitar per a desenvolupar una aplicació millor. No obstant, aquest fet no treu que la valoració personal del projecte sigui molt positiva.

7. Bibliografia

[1] Luscombe, Greenbaum, Gerstein (2001) “What is Bioinformatics? A Proposed Definition and Overview of the Field”

[2] Kitchen, Decornez, Furr, Bajorath (2004) *Docking and scoring in virtual screening for drug discovery: methods and applications*

[3] Perkins, Fang, Tong, Welsh:Structure-activity relationship methods: perspectives on drug discovery and toxicology, Environmental Toxicology and Chemistry Volume 22, Issue 8, pages 1666–1679

[4] Fukunishi, Teramoto, and Shimada (2007) “Hidden Active Information in a Random Compound Library: Extraction Using a Pseudo-Structure-Activity Relationship Model” J. Chem. Inf. Model. 2008, 48, pages 575-582

[5] http://www.molsoft.com/icmpro/faq-docking.html

[6] Berkson J (1944). "Application of the logistic function to bio-assay". *Journal of the American Statistical Association*, Chapter 39, pages 357-365

[7] http://www.ats.ucla.edu/stat/stata/dae/ologit.htm

[8] Ye, Janardan, Li (2004) Two-dimensional linear discriminant analysis

[9] http://www.statsoft.com/textbook/naive-bayes-classifier

[10] Cristianini, Shawe-Taylor (2000); *An Introduction to Support Vector Machines and other kernel-based learning methods*

[11] Friedman, J. H. (1989). "Regularized Discriminant Analysis". Journal of the American Statistical Association, page 84, pages 165–175

[12] Breiman, Leo (2001). "Random Forests". Machine Learning

[13] http://www.oracle.com/technetwork/oem/grid-engine-166852.html

[14] https://computing.llnl.gov/linux/slurm/

[15] https://vaadin.com

[16] http://tomcat.apache.org/tomcat-8.0-doc/index.html

[17] http://w3techs.com/technologies/details/ws-tomcat/all/all

[18] http://www.oracle.com/us/products/mysql/mysqlstandard/overview/index.html

[19] http://www.uniprot.org/

[20] http://www.epa.gov/med/Prods_Pubs/smiles.htm

[21] http://zinc.docking.org/

[22] http://www.eclipse.org/
[23] http://www.sublimetext.com/
[24] http://www.mysql.com/products/workbench/
[25] http://git-scm.com/
[26] http://www.slideshare.net/codento/vaadin-scalabilityslides
[27] http://aws.amazon.com/ec2/
[28] http://www.bsc.es/computer-sciences/grid-computing/comp-superscalar
[29] http://occi-wg.org/about/
[30] http://opennebula.org/
[31] http://www.medcalc.org/manual/roc-curves.php
[32] http://autonomic.ac.upc.edu/emotive/?page_id=670
[33] http://www.codessa-pro.com/descriptors/topo/balaban.htm
[34] http://www.rcsb.org

8. Annexos

8.1. Annex 1: Corbes ROC

En l'aplicació SEABED es fan servir corbes ROC(Receiver Operating Characteristic [31]) per a representar els resultats de la creació de models predictius. Aquestes corbes són una representació gràfica de l'evolució de la sensibilitat dels models classificadors contra la seva especificitat a mesura que es canvia el llindar de discriminació (valor que serveix per determinar a quin grup pertany cada element).

S'anomena sensibilitat a la capacitat dels models de classificar com a positius els elements que realment són positius (positius correctament identificats, o veritablement positius). Per altra banda, l'especificitat fa referència a la capacitat de classificar com a negatius els elements que realment són negatius (negatius correctament identificats, o veritablement negatius).

Els valors de sensibilitat i especificitat venen donats en percentatges. Idealment, un classificador binari perfecte hauria de tenir una especificat i una sensibilitat de 100%, és a dir, classificaria tots els elements correctament.

En la següent figura es mostra una corva ROC extreta de l'aplicació.

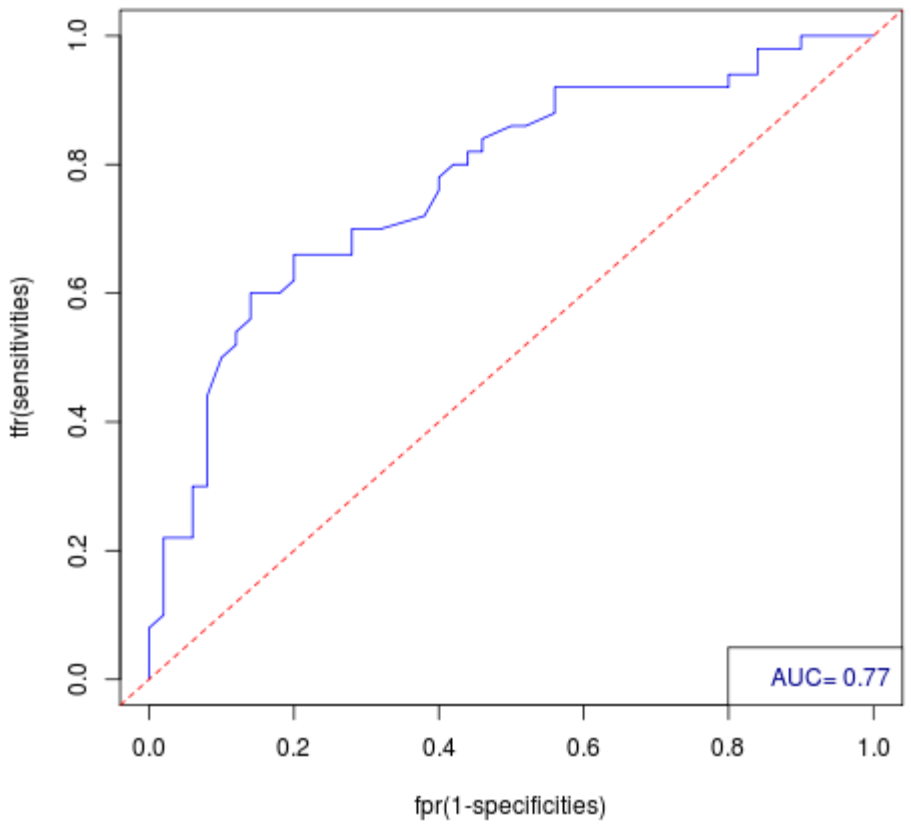


Figura : Corba AUC representant el resultat de crear un model

Per tal de comparar els models predictius s'utilitza el valor de l'àrea sota la corba (AUC en anglès). Com més proper a 1 és aquest valor, millor és el model predictiu. Per altra banda, els models predictius que presenten un valor de AUC al voltant de 0,5 són considerats com a aleatoris i per tant no són útils.

Aquest tipus de corbes són utilitzades per tal d'analitzar la validesa de proves diagnòstiques en estudis mèdics. L'escala de valors de AUC que es fan servir per als anàlisis són les següents:

[0.5, 6): Test dolent.

[0.6, 0.75): Test regular.

[0.75, 0.9): Test bo.

[0.9, 0.97): Test molt bo.

[0.97, 1): Test excel·lent.

En la figura següent s'il·lustra com es poden comparar diferents corbes ROC per a les mateixes dades d'entrenament.

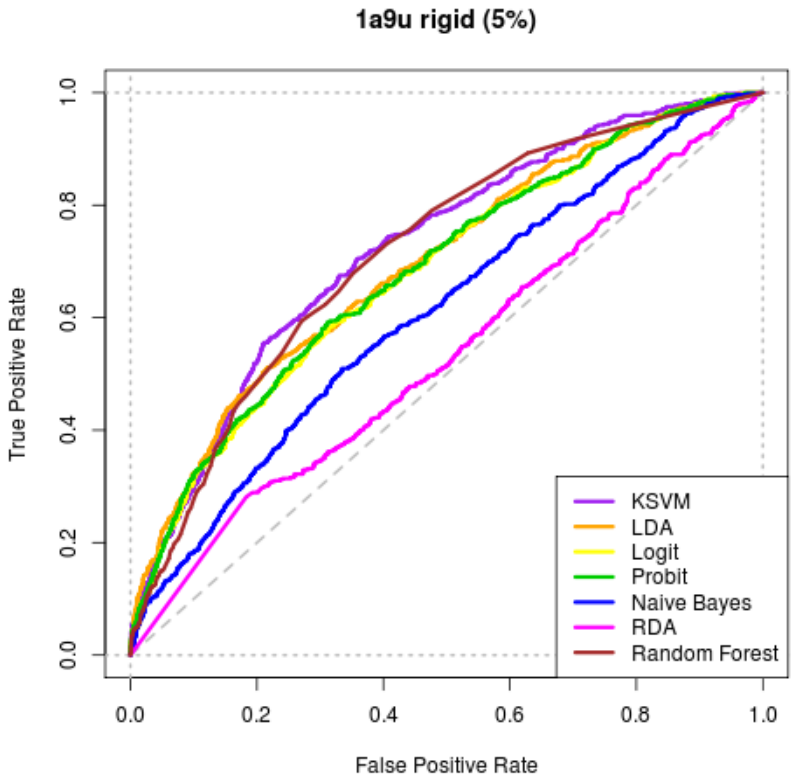


FIgura : Comparació de les corbes ROC resultants d'aplicar models creats amb els diferents algorismes de machine learning a l'aplicació SEABED

En l'exemple s'han superposat totes les corbes ROC obtingudes d'aplicar models creats a partir de les mateixes dades d'entrenament però creats amb diferents algorismes de machine learning, a les mateixes dades de test. Es pot veure com l'algorisme escollit influencia en gran manera en els resultats obtinguts posteriorment. Per comparar els models s'avaluen les AUC. En aquest cas, el millor model és el creat amb l'algorisme *Kernel Support Vector Machine (KSVM)*, mentre que el pitjor és el creat amb *Random Discriminant Analysis(RDA)*.

8.2. Annex 2: Manual d'usuari

En aquest apartat es donarà una petita guia de referència per als usuaris de l'aplicació SEABED.

8.2.1. Inici de sessió

Per utilitzar l'aplicació SEABED, cal accedir des d'un navegador web qualsevol (Firefox, Chrome, etc.) a la direcció www.bsc.es/SEABED.

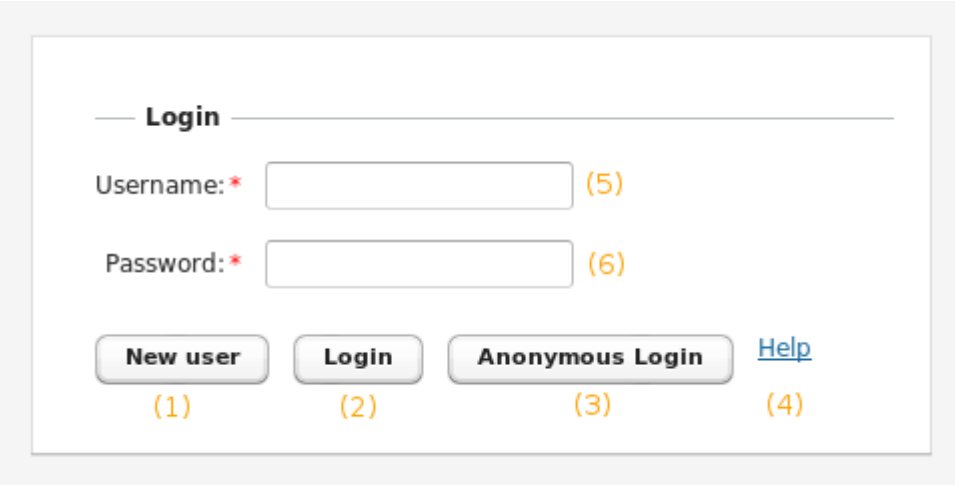


Figura 1: Formulari per entrar a l'aplicació

Es mostrarà un diàleg com l'anterior, amb les següents opcions:

- New user (1), que permetrà crear un nou usuari i registrar-lo a l'aplicació

- Login (2), que permetrà a un usuari registrat entrar a l'aplicació. L'usuari haurà de proporcionar les seves credencials: nom d'usuari (5) i contrasenya (6)

- Anonymous login (3), que permetrà entrar a l'aplicació a un usuari anònim

- Help (4), que mostrarà a l'usuari un manual d'ajuda per a l'aplicació

8.2.2. Nou usuari

Per a registrar-se a l'aplicació, cal prémer al botó "New user" ((1) en la figura 1), i omplir les dades del formulari següent:

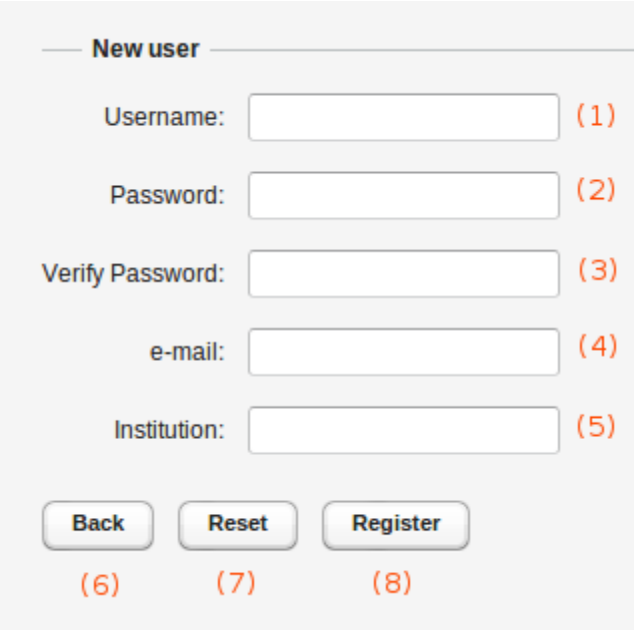


Figura: Formulari de registre a l'aplicació

Un cop omplert el formulari (camps (1) a (5)), es pot iniciar registrar l'usuari i iniciar sessió des del botó "Register" (8). Amb el botó "Reset" (7) s'esborren tots els camps de text. Per tornar enrere cap al formulari mostrat en la figura 1 es pot prémer el botó "Back"(6).

En el moment de registrar l'usuari es pot donar un dels errors següents:

- Hi ha algun camp buit
- El nom d'usuari escollit ja existeix a l'aplicació
- Les dues contrasenyes no coincideixen
- L'*email* introduït no té un format correcte

Si es dóna algun dels casos anteriors s'informarà als usuaris a través de missatges d'error, que indicaran concretament quin camp és incorrecte.

8.2.3.Pàgina principal

En la figura següent es mostra la pantalla principal de l'aplicació, que es mostra quan els usuaris (registrats o anònims) inicien la seva sessió.

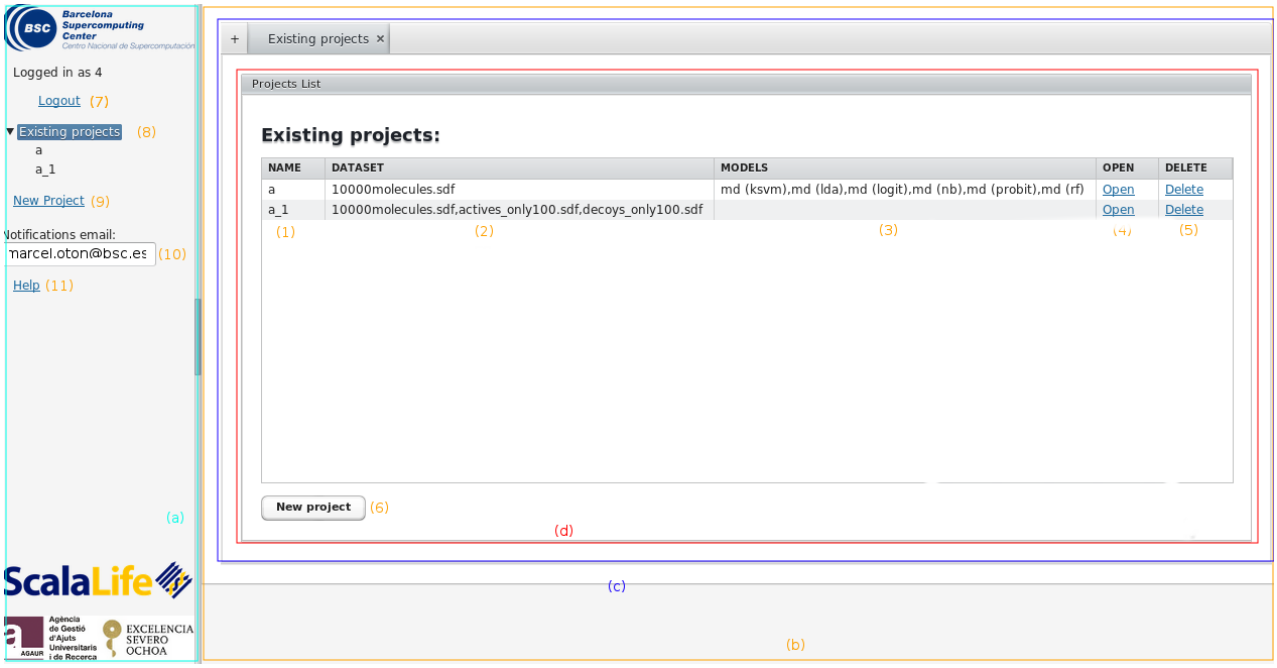


Figura: Pantalla principal de l'aplicació

La finestra de l'aplicació està dividida en dues parts per una línia vertical. La diferència principal entre les dues parts és que la part esquerra (a) és una barra fixa a l'aplicació, mentre que els components de la part dreta (b) van canviant depenent de les accions de l'usuari.

Els components de la part dinàmica s'agrupen en un *TabSheet* (c), un component del framework *Vaadin* que serveix per agrupar components per pestanyes, d'una forma semblant a com ho fan els navegadors actualment. Aquest element permet que els usuaris

no es vegin bloquejats quan els càlculs llargs s'estan executant, i puguin realitzar altres accions paral·lelament.

Per altra banda, dins de cada pestanya els components s'agrupen en panells desplegables (*Accordions* (d) en la terminologia de *Vaadin*). Gràcies a aquest component els usuaris poden tornar enrere en qualsevol dels *workflows*. Això permet als usuaris corregir possibles errors o bé recordar els paràmetres utilitzats en un pas previ.

La pàgina principal serveix per mostrar als usuaris informació sobre els seus projectes. Concretament, a la part central de la pàgina hi ha una llista on es mostren tots els projectes que ha creat l'usuari (1), amb el dataset (2) i els models (3) que formen part de cadascun. Es pot obrir un projecte (4) per continuar treballant amb les seves dades o bé es pot esborrar (5) des de la mateixa llista. Finalment, també hi ha un botó que permet crear un nou projecte (6)

Per altra banda, a la part esquerra de la pantalla hi ha una barra lateral on es mostra informació de l'usuari connectat. Des d'aquesta es pot sortir de l'aplicació (7), es mostra un arbre amb els projecte de l'usuari (8), hi ha un botó que permet crear un nou projecte (9), un camp de text on l'usuari emmagatzema un compte de correu per tal de rebre notificacions (10) i hi ha un enllaç a l'ajuda de l'aplicació (11).

8.2.4. Crear projecte

Per tal de crear un nou projecte, els usuaris poden seleccionar el botó de "New project" des de la Pàgina principal ((6) en la figura anterior) o bé des de qualsevol pàgina de l'aplicació des de la barra lateral ubicada a la part esquerra de la pantalla (10).

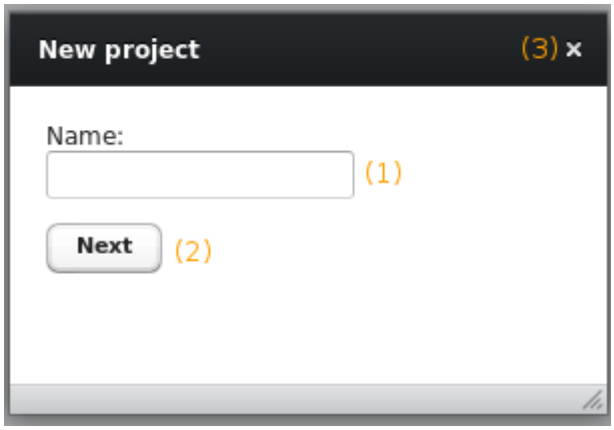


Figura: Diàleg per afegir un nom a un nou projecte

Apareixerà un diàleg on l'usuari podrà introduir el nom del projecte (1) i proseguir amb la creació d'aquest (2). Si es desitja cancel·lar la creació del projecte es pot prémer el botó situat a la part superior dreta del diàleg (3).

Posteriorment al *TabSheet* de l'usuari apareixerà una nova pantalla on podrà pujar nous fitxers per a que formin part del dataset del projecte que està creant:

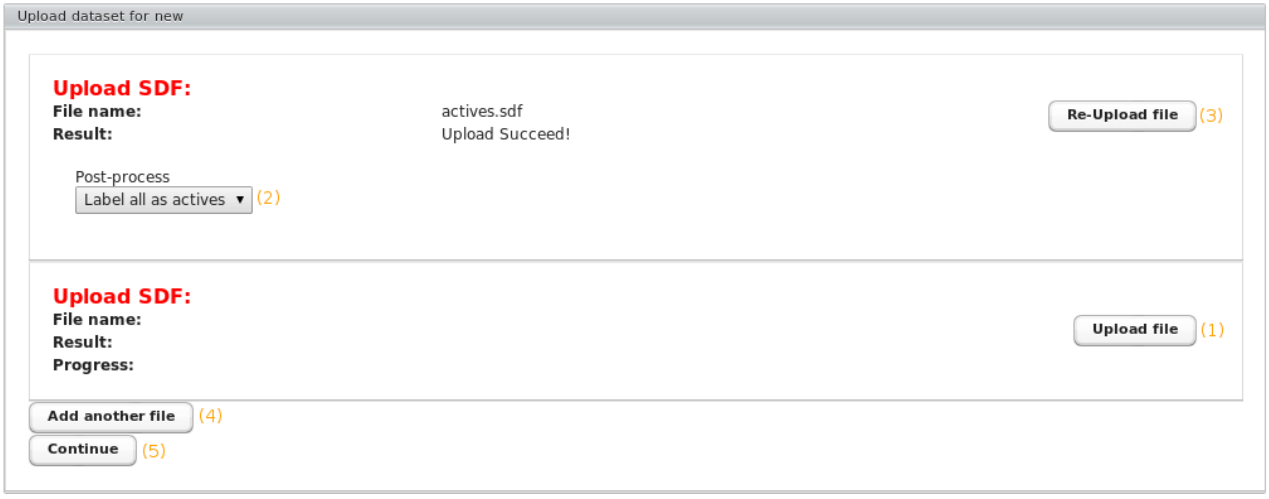


Figura: Formulari per pujar nous fitxers a un projecte

El procediment per afegir fitxers al projecte és el següent:

1. Prémer el botó “Upload file” (1) i seleccionar el fitxer que es vol pujar
2. Mitjançant la llista desplegable (2), afegir etiquetes al fitxer:
    a. “Label all as actives” marcarà totes les molècules com a actives
    b. “Label all as inactives” marcarà totes les molècules com a inactives
    c. “Add custom label” permet als usuaris afegir un parell (etiqueta, valor) a totes les molècules del fitxer
    d. “Do nothing” no modificarà el fitxer
3. Mitjançant el botó “Re-upload file” es pot pujar un fitxer sobreescrivint l’anterior
4. Per a afegir un nou fitxer, prémer el botó “Add another file” i repetir els passos 1, 2 i 3
5. Un cop s’ha pujat tots els fitxers, prémer el botó “Continue” per continuar cap a la Pàgina de gestió de projecte

8.2.5. Pàgina de gestió de projecte

Des d’aquesta pàgina es pot, per una banda, mostrar tota la informació d’un projecte, i, per l’altra, accedir a totes les funcionalitats d’aquest.

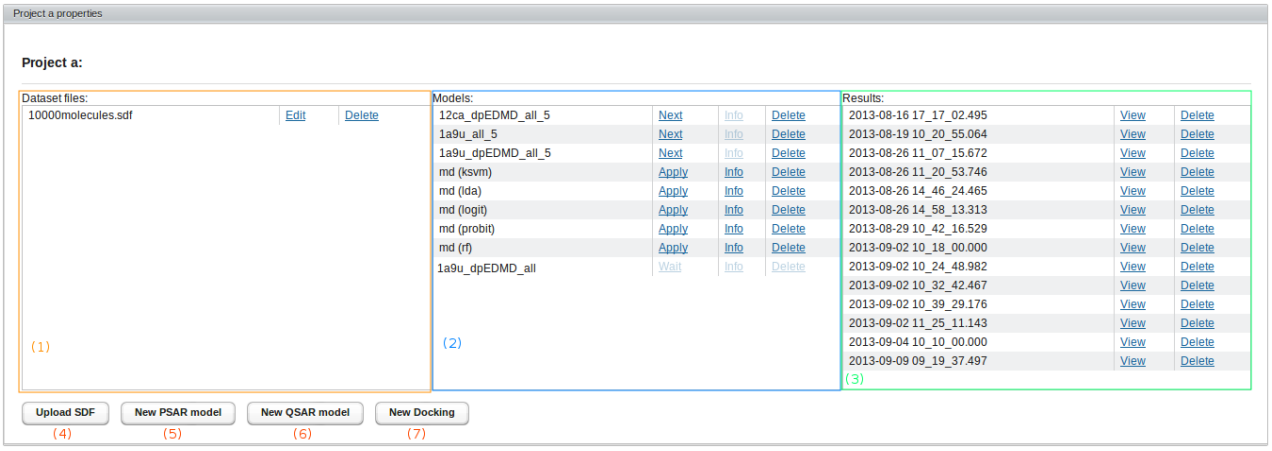


Figura: Pàgina de gestió d’un projecte

Per una banda, es mostren tres llistes diferents: dataset (1), models (2) i resultats (3). Per l'altra banda, hi ha un conjunt de botons, que permeten: pujar nous fitxers al projecte (4), crear un nou model utilitzant la metodologia PSAR (5), crear un nou model utilitzant la metodologia QSAR (6) o llençar un nou *docking* (6).

8.2.5.1. Llistat de dataset

En aquest llistat es mostren tots els fitxers que l'usuari ha pujat per a aquell projecte.

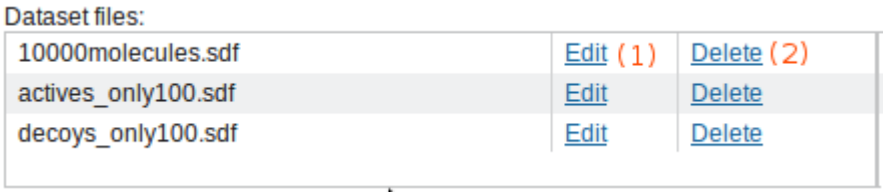


Figura: Llistat de fitxers del dataset

Al costat de cada nom del llistat hi ha dos botons que permeten modificar el fitxer (1) o bé esborrar-lo permanentment del projecte(2).

8.2.5.2. MODIFICAR FITXER DEL DATASET

Des de l'aplicació es permet eliminar molècules i modificar els fitxers que s'han pujat com a dataset d'un projecte. Cal tenir en compte que els canvis realitzats en un fitxer són irreversibles.

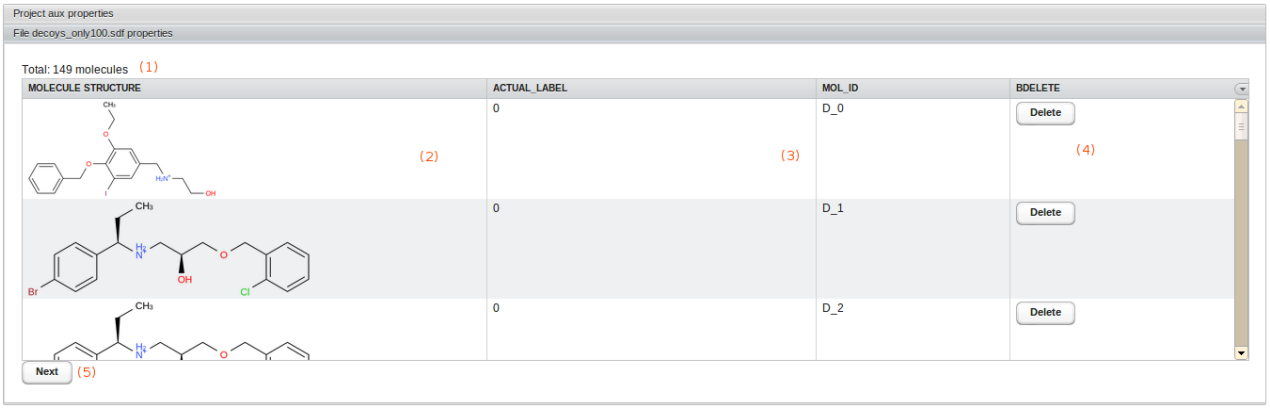


Figura : Finestra de modificació d'un fitxer de dataset

En la figura anterior es mostra la informació d'un fitxer de *dataset:* el nombre de molècules que formen part del fitxer (1), i, per a cada molècula, una representació bidimensional d'aquesta (2), tantes columnes com etiquetes hi hagi en el fitxer (3) i finalment un botó que permet esborrar la molècula del fitxer permanentment.

Un cop finalitzada l'edició del fitxer, es poden guardar els canvis realitzats i retornar a la pàgina de gestió del projecte polsant el botó "Next" (5).

8.2.5.3. Llistat de models

En aquest llistat es mostren els models del projecte (tant els que s'estan creant com els que han finalitzat) i els dockings que estan pendents de finalitzar.

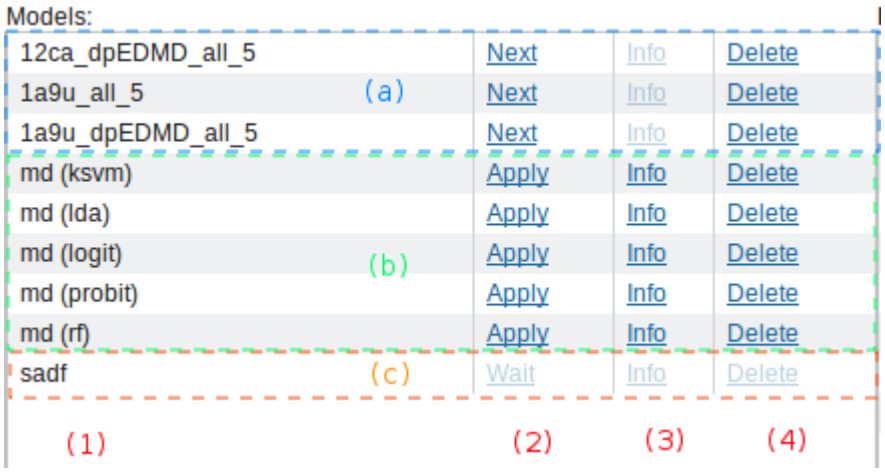


Figura : Llistat de models

En la figura 7 es mostra el llistat de models. Cada element del llistat disposa de 4 columnes. En la columna (1) es mostra el nom que l'usuari ha donat al model o docking, en la columna (2) s'informa de l'estat de l'element:

- "Apply"(b) indica que es tracta d'un model finalitzat i disponible per ser aplicat
- "Next" (a) indica que es tracta d'un *workflow* inacabat, però que es pot continuar ja que no té cap càlcul pendent

- “Wait” (c) indica que es tracta d’un *workflow* inacabat, però que no es pot continuar ja que s’està pendent de que finalitzi algun càlcul. Un cop finalitza el procés, es passa a l’estat “Next”.

La columna (3) ofereix un enllaç per consultar la informació d’un model ja creat. Aquesta columna no estarà disponible per als models que estan en construcció. Finalment, la columna (4) permet esborrar els models, sempre i quant aquests no estiguin realitzant càlculs en algun dels sistemes de cues.

8.2.5.4. INFORMACIÓ DE MODEL

Un cop ha finalitzada la creació d’un model, es poden consultar els fitxers de dataset, l’algorisme de *machine learning* (2), els paràmetres que s’han utilitzat per crear -lo(3) com la corba ROC[2](1) resultant.

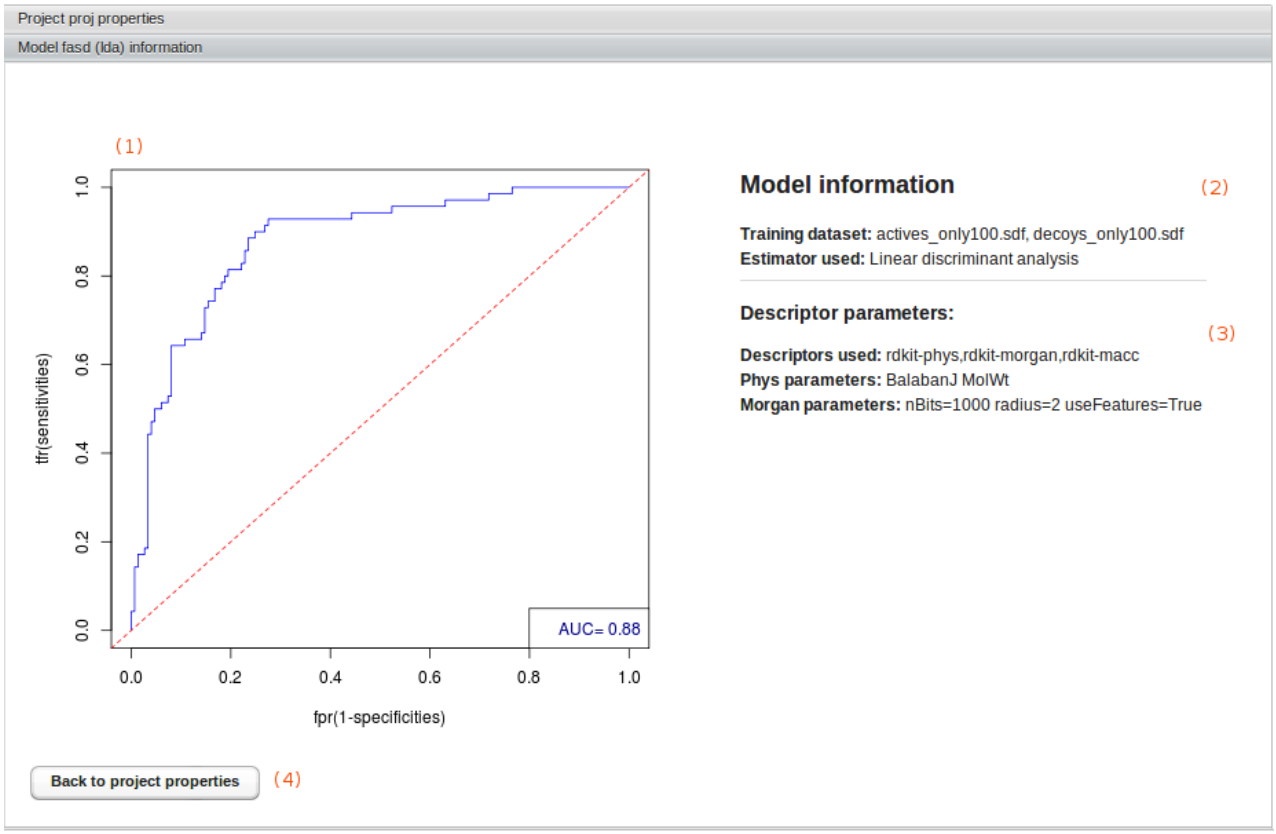


Figura : Finestra d’informació de model

8.2.5.5. Llistat de resultats

En aquesta llista es mostren els resultats obtinguts en el projecte. Hi ha tant els resultats de *dockings* com els resultats d'aplicar models. Els resultats venen identificats per l'instant de temps en què es van produir.

Figura: Llistat de resultats

Per a mostrar tota la infomació dels resultats cal accedir als resultats cal clicar al botó "View" (1). Si es vol esborrar permanentment un resultat cal clicar al botó "Delete" (2).

RESULTATS D'APLICAR UN MODEL

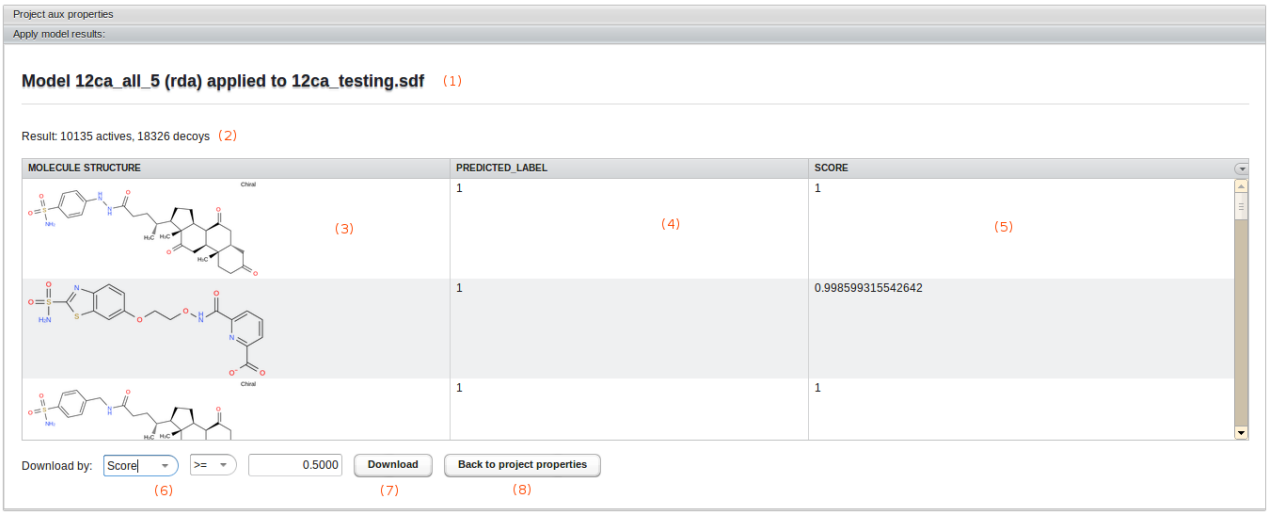


Figura : Finestra de resultats d'aplicar un model

En la figura anterior es mostra la informació resultant d'aplicar un model a unes molècules de test. Concretament, es mostra el nom del model aplicat i el nom del fitxer que conté les molècules de test (1) i el nombre de molècules que hi ha de cada tipus (2). A més a més, es mostra una taula on hi ha, per a cada molècula trobada en el fitxer de test, una representació bidimensional d'aquesta (3), el valor d'activitat predita (0 si és inactiva o 1 si és activa) (4) i finalment el percentatge d'activitat (5).

Per altra banda, es poden descarregar aquests resultats filtrant les molècules per dos valors: activitat o puntuació. El filtre per activitat permet baixar només les molècules actives, només les inactives o ambdós grups. El filtre per puntuació permet fixar un valor (entre 0 i 1) i seleccionar totes les molècules que tenen una puntuació (5) superior, inferior o igual a aquest valor. Per tal de realitzar aquest filtre s'ha d'escollir un valor de la llista desplegable (6) i posteriorment seleccionar el botó "Download" (7). Per tornar cap a la pàgina de gestió del projecte s'ha de prémer el botó "Back to project properties" (8).

8.2.5.6. RESULTATS DE DOCKING

La pantalla de resultats de docking és molt semblant a la de resultats d'aplicar un model.

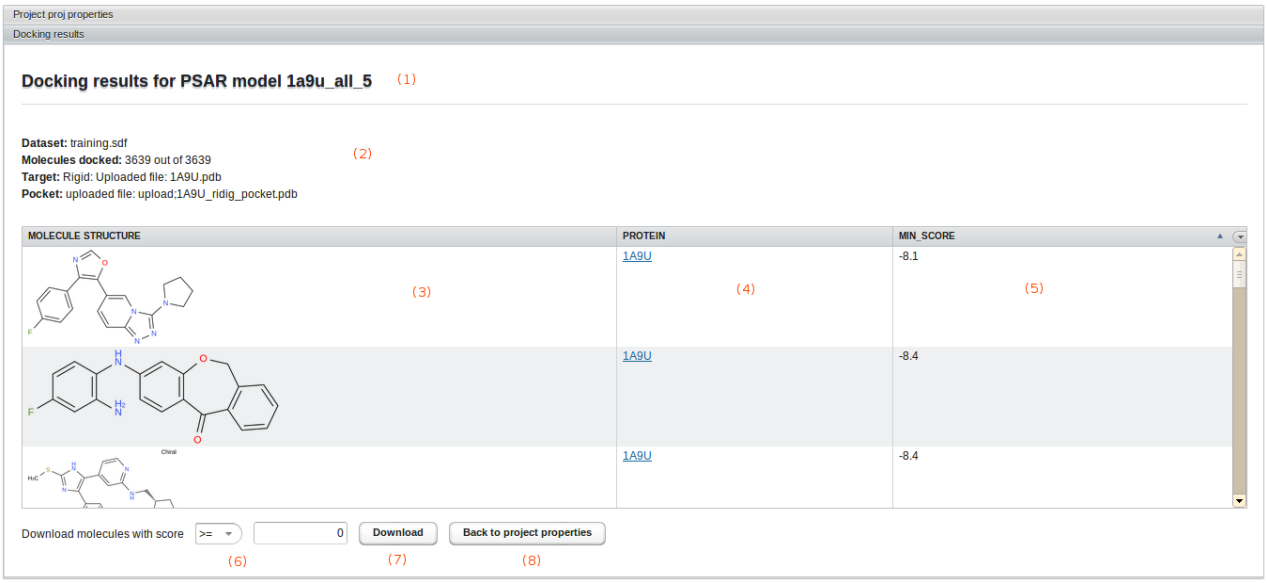


Figura : Resultats d'un docking utilitzat per crear un model amb la metodologia PSAR

En aquest cas, es mostra un títol (1) amb el nom del docking o bé del model PSAR pel qual es va fer el docking. A més, es mostra un requadre (2) amb tots els paràmetres utilitzats per a realitzar el docking: fitxers seleccionats, nombre de molècules sobre les quals s'ha fet el docking, i informació del target i el pocket utilitzats.

Posteriorment es mostra un quadre similar a l'anterior, en el que hi ha també una representació bidimensional de totes les molècules(1), un enllaç a la proteïna utilitzada per al docking(4) (el target si es tracta d'un docking rígid o bé el nom del *snapshot* si es tracta d'un *docking* elàstic) i finalment el resultat del docking amb cadascuna de les molècules (5). Si el docking és elàstic, aquest valor és el mínim obtingut de tots els *snapshots*, i a més es mostra una altra columna amb el valor mitjà dels *dockings*.

Es poden descarregar aquests resultats filtrant les molècules pel seu resultat de *docking* fent servir el desplegable (6) i polsant el botó "Download" (7). Anàlogament al cas

anterior, es pot tornar a la pantalla de gestió del projecte polsant el botó “Back to project properties” (8).

8.2.5.7. Llençar docking

Per tal de llençar un docking, els usuaris han de dirigir-se a l’opció “New docking” en la pàgina de gestió del projecte on es vol crear ((7) en la figura 5). Els passos a seguir per llençar un docking són:

8.2.5.7.1. SELECCIONAR ELS FITXERS DE DATASET

Figura: Finestra de selecció de fitxers de dataset

Seleccionar els fitxers de dataset que es vulguin fer servir per al docking de la llista de datasets disponible (1). Com a mínim un dels fitxers ha d'estar seleccionat per poder continuar. Posteriorment polsar el botó "Next" (2).

8.2.5.7.2. SELECCIONAR LES MOLÈCULES PER AL DOCKING

En aquesta finestra es mostra una taula amb totes les molècules que s'han trobat en els arxius seleccionats anteriorment. Per a cada molècula, es mostra la seva representació bidimensional (1) i les seves etiquetes (2). Si no es desitja que el *docking* sigui sobre totes les molècules, es pot proporcionar el nombre desitjat a (3). El nombre de molècules ha de ser com a mínim 1 i com a màxim el nombre total de molècules trobades.

Figura: Finestra de selecció del nombre de molècules per al docking

Un cop introduït el nombre de molècules sobre el que es farà el docking, cal polsar el botó "Next" (4) per continuar.

8.2.5.7.3. SELECCIONAR DIANA

Un cop s'han seleccionat les molècules, cal escollir quina serà la diana terapèutica sobre la qual es farà el docking. L'aplicació ofereix la possibilitat de realitzar dockings elàstics o rígids.

Figura: Finestra de selecció de diana

Si es desitja realitzar un docking rígid (sobre l'estructura original d'una proteïna) cal seleccionar l'opció "PDB" com a "Target" (1). En canvi, si es vol realitzar un *docking* elàstic (sobre els *snapshots* d'una estructura en una trajectòria), cal seleccionar l'opció "MD" (Molecular Dynamics).

8.2.5.7.3.1. DOCKING RÍGID

L'aplicació ofereix dues opcions per realitzar un *docking* rígid: l'usuari pot pujar un fitxer que contingui una referència de la proteïna que vol utilitzar com a diana, o bé pot seleccionar una proteïna present a la base de dades del Protein Data Bank [34].

Per tal de pujar un fitxer, s'ha de seleccionar l'opció "File" en el camp de "Source" (1), i posteriorment pujar un fitxer .pdb que contingui l'estructura de la proteïna.

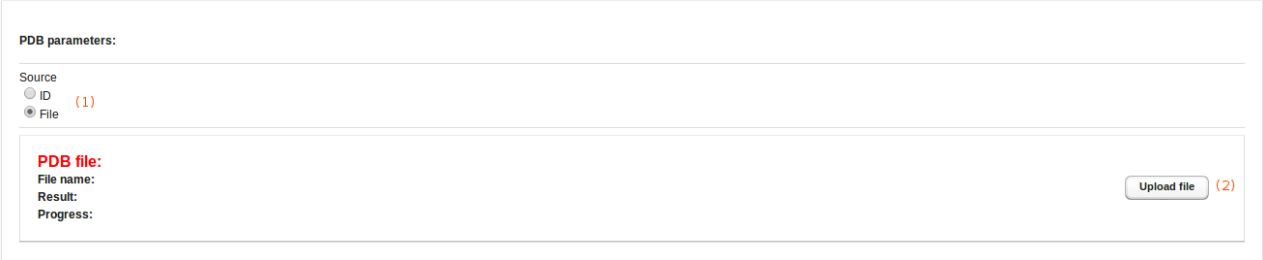


Figura: Finestra per seleccionar un docking rígid pujant un fitxer que conté l'estructura a utilitzar

Si en canvi es vol utilitzar una estructura del Protein Data Bank (PDB), s'ha de seleccionar l'opció "ID" del camp "Source" (1) i introduir el codi de la proteïna utilitzat en el PDB (2).

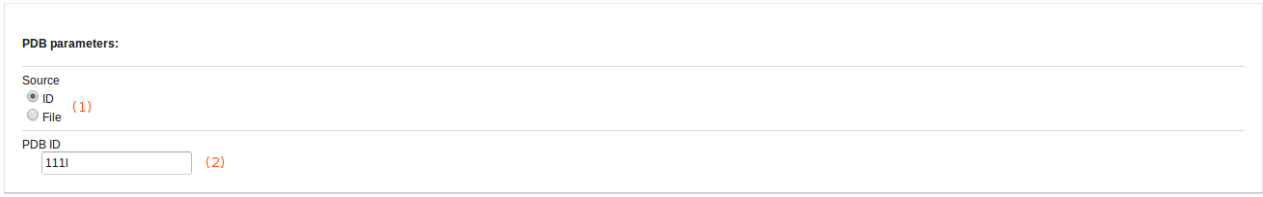


Figura : Finestra per seleccionar l'estructura d'un docking rígid on es fa servir l'estructura 111L del Protein Data Bank.

Un cop s'ha seleccionat l'estructura seguint qualsevol dels mètodes anteriors, cal polsar el botó "Next" per continuar.

8.2.5.7.3.2. DOCKING ELÀSTIC

Per realitzar un *docking* elàstic cal pujar un fitxer que contingui l'estructura de referència de la proteïna ((1), en un fitxer .pdb) i una trajectòria d'aquesta estructura ((2), en un fitxer .netcdf).

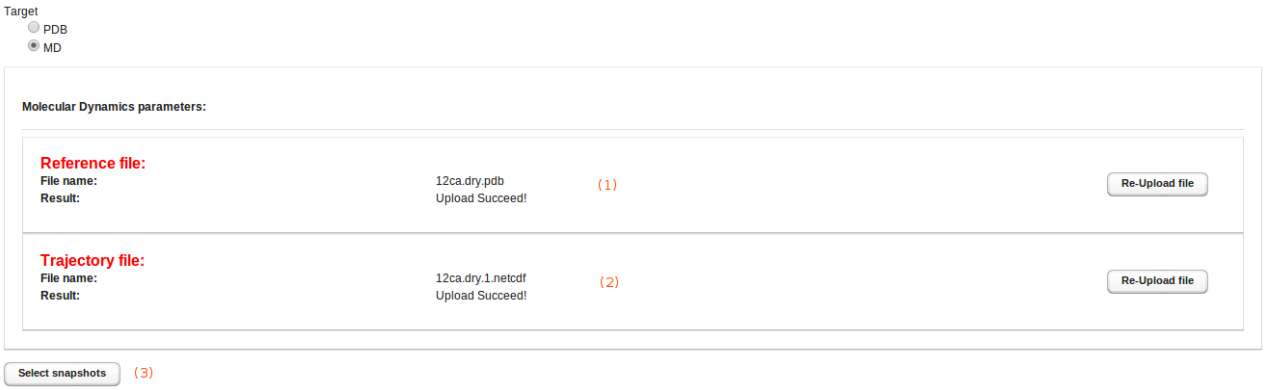


Figura : Finestra de selecció d'un target i una trajectòria per realitzar un docking elàstic

Un cop pujats els dos fitxers, cal polsar el botó "Select Snapshots" (3) per tal de seleccionar els *snapshots* que es faran servir per al docking. Quan comensi l'execució del programa de selecció de *snapshots* es modificarà el botó per un *spinner*. Un cop finalitzi aquesta execució, es mostrarà un botó "Next" que cal polsar per continuar.

8.2.5.7.4. SELECCIONAR POCKET

L'últim pas abans de llençar un docking és seleccionar el *pocket* de la estructura. L'aplicació SEABED té tres opcions diferents per seleccionar un *pocket*: pujant un fitxer, seleccionant-lo manualment o seleccionant-lo automàticament.

8.2.5.7.4.1.PUJAR POCKET DES D'UN FITXER

Cal seleccionar l'opció "Upload pocket"(1) i posteriorment afegir un fitxer .pdb que contingui l'estructura del *pocket* mitjançant el botó "Upload file" (2).

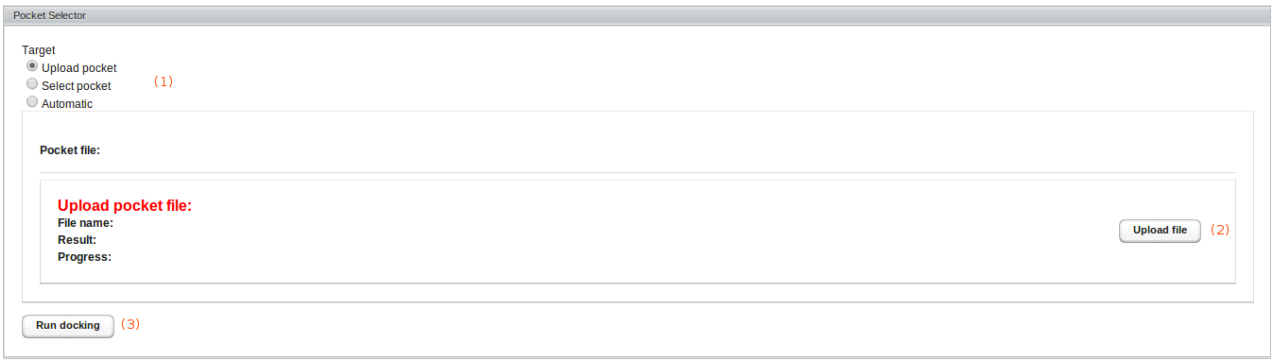


Figura : Selecció de pocket des d'un fitxer

8.2.5.7.4.2. SELECCIONAR POCKET MANUALMENT

Un altre mètode de selecció del pocket consisteix en seleccionar manualment els residus que el conformen. Per a utilitzar aquest mètode, cal seleccionar l'opció "Select pocket" (1) i posteriorment marcar les caselles dels residus que es volen seleccionar de la llista (2)

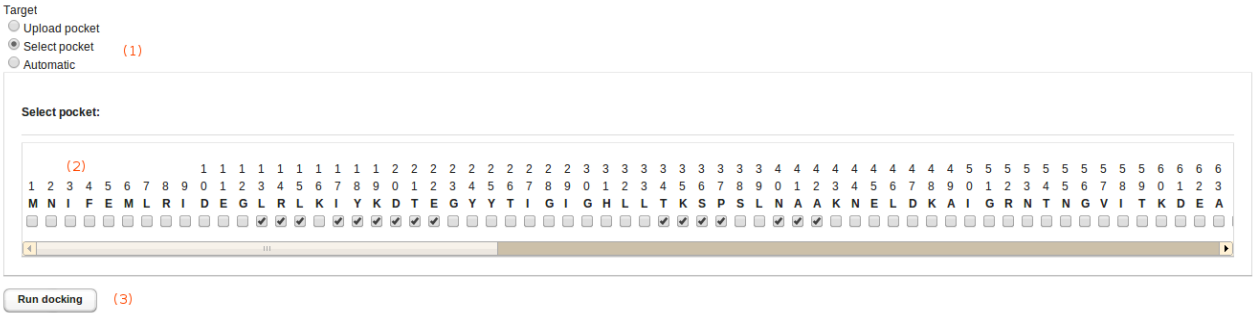


Figura : Selecció de pocket manual

8.2.5.7.4.3. SELECCIONAR POCKET AUTOMÀTICAMENT

Finalment, l'aplicació ofereix un mètode que calcula els *pockets* automàticament. Si es vol utilitzar, només cal seleccionar l'opció "Automàtic" (1).

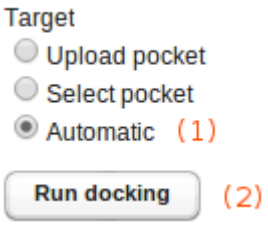


Figura : Selecció de pocket automàtica

Un cop s'ha seleccionat un dels tres mètodes anteriors, només cal polsar el botó "Run docking" per començar el docking. Es mostrarà un *spinner* mentre es realitzen els càlculs. S'estima que es tarda una mitjana de 90 segons per cada *docking*. Si es tracta

d'un *docking* elàstic, cal multiplicar el temps pel nombre de *snapshots,* 5 en el cas de l'aplicació.

Quan finalitzi el processament del docking, es mostrarà una finestra igual a la mostrada en l'apartat anterior "Resultats de docking".

8.2.5.8. Crear model QSAR

Per crear un model amb la metodologia QSAR (tenint un registre previ de l'activitat de les molècules d'entrenament), cal seleccionar l'opció "New QSAR model" des de la pàgina de gestió del projecte on es vulgui crear.

8.2.5.8.1. LLISTAT DEL DATASET

Es tracta de la mateixa finestra que en el cas de llençar *docking.* Cal seleccionar els fitxers del dataset que formaran part de les dades d'entrenament del model.

8.2.5.8.2. LLISTAT DE MOLÈCULES D'ENTRENAMENT

Aquesta és una finestra informativa, on es mostren el nombre de molècules de cada grup (actius i inactius) (1) i una taula on es mostren representacions bidimensionals de les molècules (1) així com les seves etiquetes (4) i (5) i el grup al qual pertanyen (3).

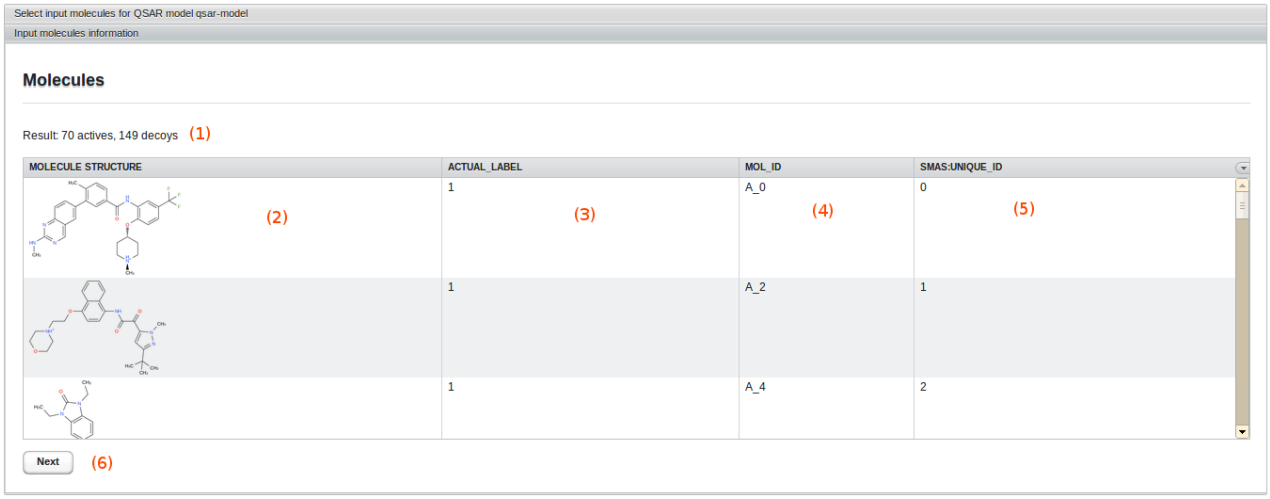


Figura : Finestra d'informació de les molècules d'entrenament

Per continuar amb la creació del model cal polsar el botó “Next” (6).

8.2.5.8.3. CALCULAR DESCRIPTORS

El següent pas consisteix en calcular els descriptors de les molècules d’entrenament del model.

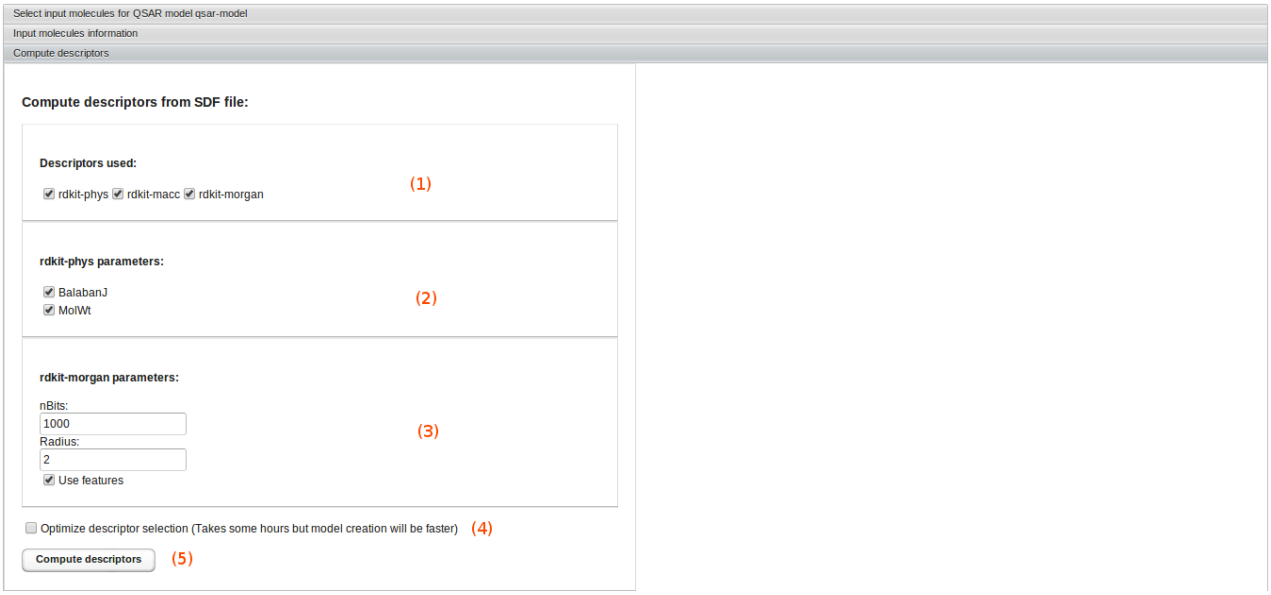


Figura : Finestra de càlcul dels descriptors

Primerament, cal seleccionar quin tipus de descriptors es volen calcular (1):

- Físics (rdkit-phys)
- MACC *keys* (rdkit-macc)
- *Morgan fingerprints* [32] *(rdkit-morgan)*

Posteriorment, cal seleccionar els paràmetres dels descriptors, si escau:

- Es pot escollir quins dels descriptors físics es calcularan: pes molecular (*MolWt*) i índex J de Balaban [33] (BalabanJ)
- Es poden canviar els paràmetres per a calcular els *Morgan fingerprints*: número de bits (nBits) i el radi (radius)

L'aplicació ofereix la possibilitat de fer una purga posterior dels descriptors. Aquesta purga esborrarà els descriptors que no influeixen en l'activitat. Això fa que el temps de càlcul dels descriptors augmenti considerablement, encara que posteriorment la creació del model serà molt més ràpida, ja que es disposarà només dels descriptors discriminants. Per a escollir aquesta funcionalitat cal seleccionar l'opció "*Optimize descriptor selection"* (4).

Per a calcular els descriptors cal polsar el botó "Compute descriptors" (5). Es mostrarà un *spinner* mentre s'estigui executant el procés. Un cop finalitzat apareixerà un botó "Next" que caldrà polsar per continuar.

8.2.5.8.4. MODIFICAR FITXER D'ENTRENAMENT

Aquesta pàgina és anàloga a la de modificar els fitxers de dataset. No obstant, en el cas dels fitxers de dataset els canvis eren permanents, i en aquesta finestra els canvis només afectaran la creació del model, i, posteriorment, es desfaran els canvis.

8.2.5.8.5. SELECCIONAR ALGORISME DE *MACHINE LEARNING*

L'últim pas en la creació d'un model utilitzant la metodologia QSAR consisteix en l'elecció d'un algorisme de *machine learning*. Cal escollir el desitjat d'una llista (1) i , depenent de l'escollit proporcionar els seus paràmetres (2). Finalment, cal polsar el botó "Create model" (3) per començar amb la creació del model. Un cop finalitzi la creació del model es mostrarà la finestra "Informació de model" vista anteriorment.

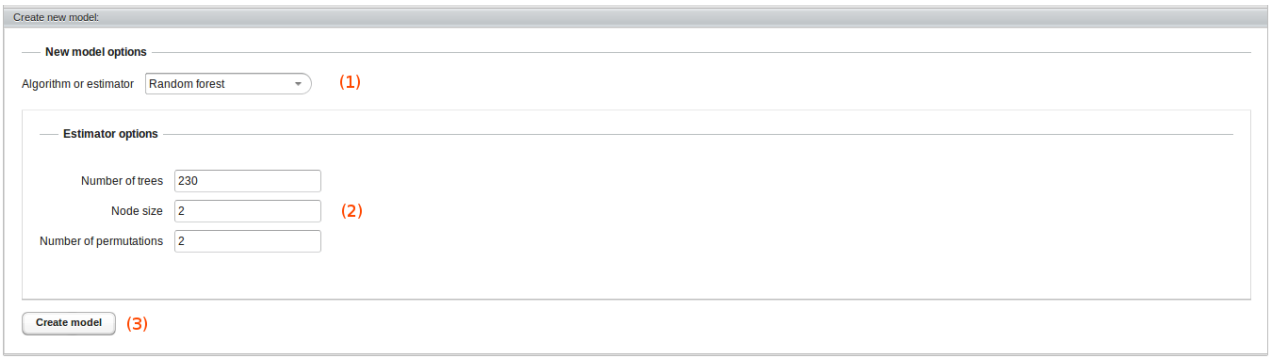


Figura : Finestra d'elecció d'algorisme de machine learning per a la creació d'un model predictiu

L'aplicació ofereix la possibilitat d'utilitzar tots els algorismes per separat sobre les mateixes dades d'entrenament. Això permet que no s'hagin de repetir tots els passos anteriors si només es vol utilitzar un algorisme diferent. Per a poder utilitzar aquesta funcionalitat s'ha de seleccionar l'opció "All" en el desplegable (1). En aquest cas, l'aplicació esperarà a que finalitzin tots els models i llavors es tornarà a la pantalla de gestió del projecte. A la llista de models creats hauran aparegut 7 models amb el nom del model creat, amb la diferència que tindran afegit al final del nom l'algorisme de *machine learning* utilitzat. En la següent figura s'il·lustra aquest cas:

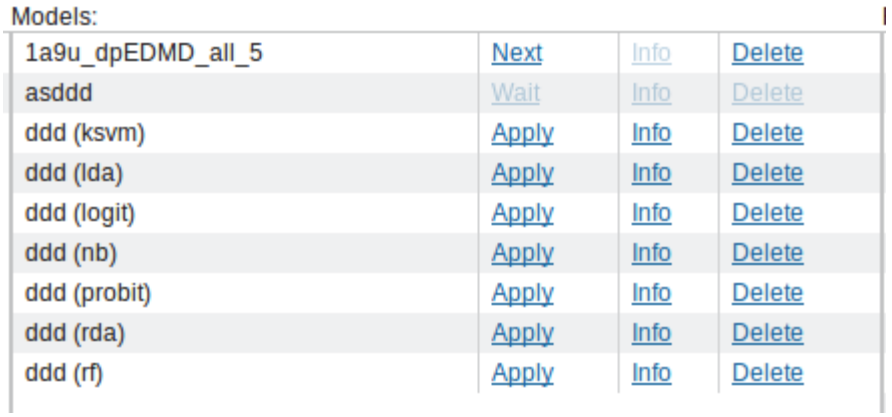


Figura : Resultat de crear un model amb tots els algorismes de machine learning

8.2.5.9.Crear model PSAR

La metodologia PSAR és un híbrid docking/QSAR. En aquest cas no es disposa d'informació prèvia sobre l'activitat de les molècules, sino que primer es realitza un docking sobre les molècules i posteriorment es marca un conjunt de les que han obtingut millor resultat com a pseudo-actives.

Per accedir a aquesta funcionalitat, cal seleccionar el botó "New PSAR model" des de la finestra de gestió del projecte on es vulgui crear. Els passos següents són els mateixos que els de llençar un docking.

8.2.5.9.1. ASSIGNACIÓ DE PSEUDOACTIVITAT A LES MOLÈCULES

Un cop es disposa dels resultats de docking, el següent pas és determinar quin percentatge de molècules es marcaran com a pseudoactives.

Figura: Panell on es permet seleccionar les molècules pseudo-actives

En aquesta finestra es mostren els resultats d'un docking. Primerament, es disposa d'una breu descripció dels paràmetres utilitzats en el docking(1). A més, hi ha una taula amb els resultats obtinguts per a cada molècula. Si el docking realitzat prèviament és rígid, es mostra el resultat obtingut (2). En canvi, si el docking és elàstic, es mostren dues columnes de resultats, el millor resultat obtingut (2) i la mitjana de tots els resultats(3). Cal recordar que el docking elàstic consistia en seleccionar un nombre de *snapshots* i realitzar un *docking* rígid amb cadascun d'ells. En aquest context, el millor resultat és el mínim dels resultats de tots els *snapshots* i l'altre es la mitjana de tots aquests resultats.

Posteriorment cal seleccionar quin és el percentatge de molècules que es vol marcar com a pseudo-actiu. Cal introduir el valor (entre 1 i 100) de molècules que es volen marcar com a pseudoactives. Les molècules es poden ordenar pel seu resultat de dockingo pel valor mitjà (en cas de tractar-se d'un docking elàstic). Això es pot seleccionar des del desplegable (5).

Un cop determinat el percentatge, cal seleccionar el botó “Next” (6) per continuar amb la creació del model. Els següents passos són iguals que els descrits per a la creació d’un model QSAR, a partir del pas “Llistat de molècules d’entrenament”.

8.2.5.10. Aplicar model

Un cop creats, els models es poden aplicar sobre molècules de *test* per tal de predir la seva activitat. Per aplicar un model, independentment de la metodologia utilitzada per a crear-lo, cal seleccionar-lo de la llista de models disponible a la pàgina de gestió del seu projecte.

8.2.5.10.1. SELECCIONAR MOLÈCULES DE TEST

El primer pas és seleccionar les dades de test sobre les quals s’aplicarà el model. Aquestes dades es poden seleccionar des de dues fonts: pujant-les des d’un fitxer o bé utilitzant un conjunt de molècules de la base de dades Zinc$_{20}$.

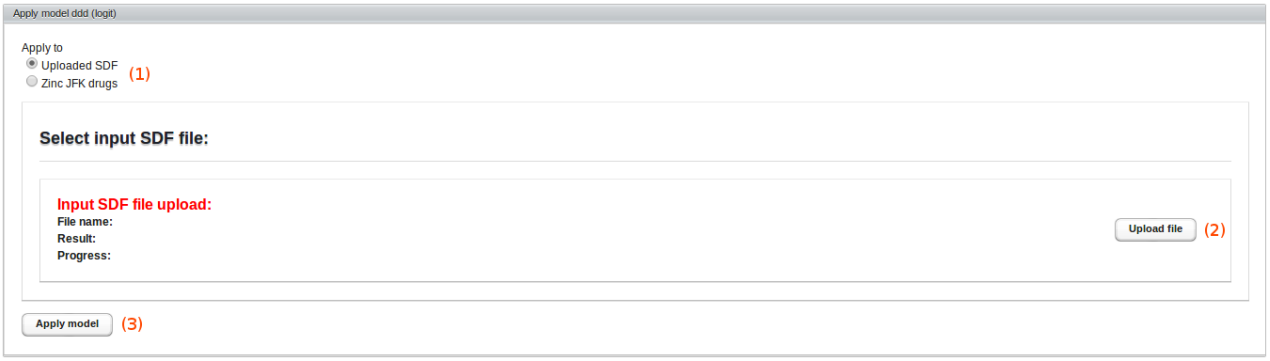


Figura : Finestra de selecció de dades de test sobre les que s’aplicarà un model

Si es vol pujar les molècules de test des d’un fitxer, cal seleccionar l’opció “Upload SDF” en el camp “Apply to” (1). Posteriorment, s’ha de pujar el fitxer fent servir el botó “Upload file” (2) i seleccionant la ruta de l’arxiu. Un cop pujat el fitxer, només cal polsar el botó “Apply model”(3) i esperar a que finalitzi el procés.

Per altra banda, si es vol aplicar un model sobre dades del subset JFK del zinc, cal seleccionar l'opció "Zinc JFK drugs" del camp "Apply to" (1) i posteriorment polsar el botó "Apply model" (3).

Un cop finalitzat el procés, es mostrarà la finestra de "Resultats d'aplicar un model", on l'usuari podrà veure i descarregar quins han sigut els valors obtinguts per a cadascuna de les molècules de *test*.